Information Sciences 647 (2023) 119502

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Local causal structure learning for streaming features

Dianlong You [a,b], Siqi Dong [a,b], Shina Niu [a,b], Huigui Yan [a,b], Zhen Chen [a,b], Shunfu Jin [a,b], Di Wu [c,*], Xindong Wu [d]

[a] School of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei 066004, China
[b] The Key Laboratory for software engineering of Hebei Province, Yanshan University, Qinhuangdao, Hebei 066004, China
[c] College of Computer and Information Science, Southwest University, Chongqing 400715, China
[d] Research Center for Knowledge Engineering, Zhejiang Lab, Hangzhou 311121, China

ARTICLE INFO

Keywords:
Bayesian network
Local causal structure learning
Streaming features
Approximate Markov Blanket

ABSTRACT

Local causal structure learning (LCSL) is the process of discovering and distinguishing direct causes (parents) and direct effects (children) of a given target variable (T) without learning a global causal structure. However, state-of-the-art LCSL algorithms can only address static feature space, while ignoring the dynamic changes of feature space with streaming features. For high dimensional data, existing methods fail to efficiently distinguish causality between features and the target variable. To address these issues, we propose Local Causal Structure Learning for Streaming Features ($LCSL_{SF}$). $LCSL_{SF}$ comprises two sequential steps, as it first dynamically mines causal features to construct an approximate Markov Blanket (aMB) of the target variable. It proceeds by filtering irrelevant and redundant features, retaining causal features as much as possible from streaming features. Second, it learns local causal structures by mining and splicing the V-structure of feature nodes in batches from aMB of the target variable. On benchmark Bayesian networks with the number of variables ranging from 20 to 724, $LCSL_{SF}$ achieves a significantly better performance than its rivals. We demonstrate its effectiveness by conducting a real-world case study on causal discovery in medical diagnostics. Code is released at https://github.com/youdianlong/LCSLSF.

1. Introduction

Cause discovery [1–3] is a process of mining causality from observational data by strictly distinguishing "cause" and "effect" from features and the target variable. This plays an irreplaceable role in revealing causal mechanisms and guiding intervention behavior. Historically, learning causality was studied in various key domains, including medical science, economics, epidemiology, and meteorology [4]. In recent years, learning valid causal mechanisms from dynamic observable data has been a research hotspot in the fields of machine learning and artificial intelligence [5]. Specifically, learning causality attracted considerable attention in disease diagnosis and treatment [6,7], sequencing biochemistry in bioinformatics [8,9], inference of public opinion in the dynamic social system [10], sentiment analyzes of constantly updated Twitter [11], and others. From the above scenarios, the following characteristics are summarized: 1) the feature space is typically high dimensional, which makes it impractical to learn the global causal structure; 2) features arrive dynamically and fleetingly one by one, such that mined causality is likely to change over time; 3) numerous irrelevant and redundant features result in extremely high computational complexity in causal structure learning. As

* Corresponding author.
E-mail addresses: youdianlong@sina.com (D. You), wudi.cigit@gmail.com (D. Wu).

https://doi.org/10.1016/j.ins.2023.119502
Received 3 February 2023; Received in revised form 28 June 2023; Accepted 8 August 2023

Available online 15 August 2023

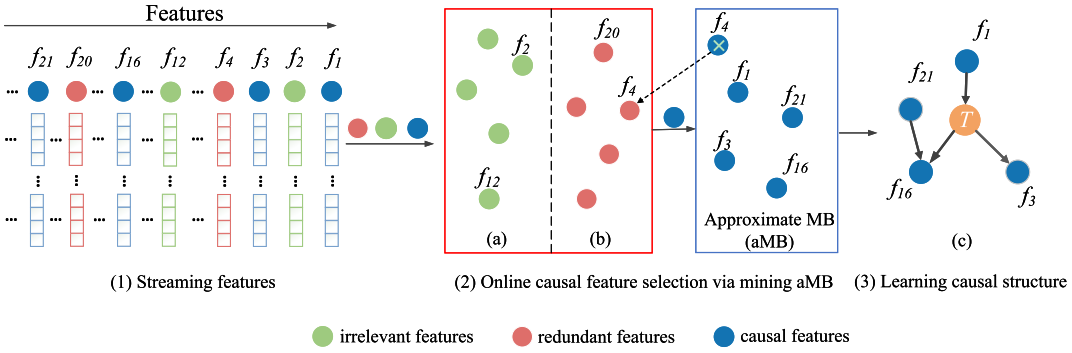


Fig. 1. The Road map of causal structure learning for streaming features. (a) online irrelevant feature filtering; (b) online redundant feature filtering.

shown in Fig. 1, we need to constantly discard irrelevant/redundant features and obtain causal features from streaming features with the continuous arrival of new features (Fig. 1 (2)), then utilize causal features to construct the local causal structure (Fig. 1(3)). Considering the completeness of generating causal structure (CS), learning CS is not suitable for synchronization with online causal feature selection. Moreover, the cause and effect we consider are often local. For the target variable, most features are irrelevant or redundant. To identify the causal relationship in features, a causal model must be established. The Bayesian Network (BN) [12] is one of the most commonly used causal models. However, in the above scenarios, exploring the causality between features in dynamic feature space poses a significant challenge.

Existing algorithms can handle the causal structure learning problem, e.g., global causal structure learning (GCSL), local causal structure learning (LCSL), and dynamic causal skeleton learning (DCSL). However, it is formidable to simultaneously learn the causal structure for streaming features. More specifically, 1) GCSL mainly learns the causal relationship of all features to obtain a complete Bayesian Network (BN), such as MMHC [13], DAG-GNN [14], and NOTEARS [15]. However, in real-world applications where only the causality of the target variable is of interest (e.g., the direct causes of "Tuberculosis or cancer" are Tuberculosis and Lung cancer, and the direct effects are X-ray results and Dyspnea, as shown in Fig. 2). In particular, when the data increases exponentially, GCSL imposes incalculable time and space costs, leading to NP-hard problems [16,17]; 2) LCSL is proposed to overcome the shortcomings of GCSL, in which only the direct causes (parents) and direct effects (children) of the target variable are discovered and learned. The existing LCSL mainly includes two general steps: first, learning the PC or MB of the target variable and constructing the local structure skeleton between the target variable and the features in the PC or MB. Second, determining the causal relationship between PC and the target variable until the parents and children of the target variable are distinguished, such as PCD-by-PCD [18], MB-by-MB [19], CMB [20], ELCS [21], and LCS-FS [22]. However, in real scenarios where data is dynamically changing in real-time and features arrive as streams, these algorithms cannot dynamically capture streaming features; 3) DCSL can address streaming features by online causal feature selection, which processes new incoming features in real-time with a fixed number of samples /instances [23]. However, DCSL mines a subset of features as a causal skeleton, and the causal direction between features, such as CDFSF [24], OSFS [25], and OCFSSF [26] cannot be constructed. For the CDFSF [24] algorithm to discover causal relationships from streaming features, but which also only learns parents and children. The OSFS [25] algorithm uses non-/multi-conditional independence to obtain MB of the target variable with relevant and non-redundant features from streaming features in real-time. Nevertheless, OSFS partly belongs to causal feature selection due to only learning PC and lack of spouse nodes. Meanwhile, the runtimes of the OSFS algorithm grow exponentially with the increasing of redundant features. SAOLA [27] algorithm is a non-causal feature selection algorithm that retains redundant features to obtain a larger set of features. OCFSSF [26] is an online causal feature selection algorithm that mined MB includes PC and spouse nodes, but the spouses are incomplete under certain conditions. However, the above online causal feature selection algorithm can only obtain PC or partial MB of the target variable. From the above, learning more complete MB in both dynamic and static feature spaces is crucial for LCSL.

Motivated by the above issues, we propose a Local Causal Structure Learning for Streaming Features ($LCSL_{SF}$) method using conditional independence tests to discover causality between features and the target variable, and constructing local causal structures. More specifically, the main idea of the $LCSL_{SF}$ is two-fold: 1) considering the dynamic feature space, streaming features arrive dynamically and are fleeting. It is impractical to mine an accurate Markov Blanket. Therefore, we propose to mine an approximate Markov Blanket (aMB) through the filtering mechanism to discard irrelevant and redundant features and retain causal features as much as possible; 2) in view of the performance and complexity of causal structure learning, we take the target variable as the core, and limit that the features that come only from the aMB. Then, we mine and splice the V-structure of feature nodes iteratively in the aMB to construct the local causal structure of the target variable to avoid the disturbance of irrelevant/redundant features and reduce the computational cost. The process faces the following critical challenges: 1) how to mine the aMB of the target variable from streaming features in real-time; 2) how to accurately distinguish between direct causes and effects of the target variable from aMB; 3) how to mine local causal structure for the target variable under the dynamic feature space.

The main contributions of this study are summarized as follows.

1) We introduce a term of approximate Markov Blanket (aMB) to learn PC, and spouses of a target variable in the dynamic feature space. Further, we propose an online algorithm of OL_{aMB} for streaming features by filtering non-PC and non-spouses as far as possible through the judgment of irrelevance and redundancy between streaming features and the target variable.

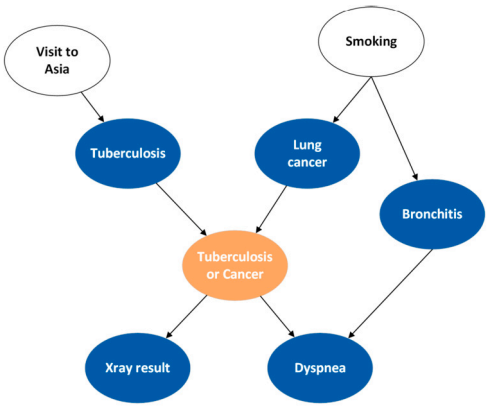


Fig. 2. MB of "Tuberculosis or Cancer" comprises the nodes of "Tuberculosis" and "Lung cancer" (parents), "Xray result" and "Dyspnea" (children), and "Bronchitis" (spouse).

2) This paper studies a novel problem of learning local causal structures from the dynamic feature space by mining an aMB of the target variable. The $LCSL_{SF}$ algorithm is proposed by mining and splicing the V-structure of feature nodes in the aMB to construct the local causal structure of the target variable.

3) Experiments are conducted extensively on thirteen benchmark BNs to demonstrate that $LCSL_{SF}$ exhibits effectiveness compared with five existing state-of-the-art algorithms for LCSL in static space, and evaluates OL_{aMB} performance on synthetic/real datasets.

The remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 introduces preliminary information, including important notations, definitions, and the $LCSL_{SF}$ framework. Section 4 describes the algorithm in detail. Section 5 reports our experimental results. Finally, the conclusions of this study are summarized in Section 6.

2. Related work

2.1. Online causal feature learning

Feature selection is a common dimensionality reduction technique in the machine learning community [28,29], which selects an "optimal feature subset" by eliminating irrelevant and redundant features and reducing the computational complexity [30]. Although feature selection reduces the dimensionality of the data and improves the prediction ability, it cannot infer the causal relationship between features. Therefore, learning causal features has become a key topic [31,3], and numerous algorithms are proposed, such as IAMB [32], HITON-MB [33], EEMB [34], and BAMB [35]. However, the above algorithms are not suitable for dynamic and high-dimensional environments.

Online causal feature learning from streaming features has been proposed [25,36,37] to address these shortcomings. To represent the causal relationship between features in the streaming data scenario, Yu et al. proposed the CDFSF algorithm and S-CDFSF [24] algorithms for causal discovery through BN. The above methods of causal feature learning only deal with novel sequentially added features, but do not evaluate the redundancy of the selected features after the addition of new ones. Wu et al. formally presented the concept of streaming features to model this dynamic nature of the feature space and presented OSFS and Fast-OSFS [25] to learn PC of the target variable by selecting strongly relevant and non-redundant features on the fly. Yu et al. presented the SAOLA [27] algorithm and designed an online pairwise comparison method for high-dimensional feature sets, which uses mutual information to compare features two-by-two and only partly filter redundant features. To further capture the causal features in the dynamic feature space, You et al. designed an online causal feature selection algorithm named OCFSSF [26], which mines MB containing the PC and spouses in real-time, and uses a learning method that interleaves PC and spouses. Resultingly, the mined MB misses a few spouse nodes and cannot distinguish between the parents and children of the target variable.

To summarize, causal feature selection mainly focuses on mining the MB of the target variable. Among the existing methods, some can handle data with fixed feature spaces, such as IAMB, HITON-MB, EEMB, and BAMB; Others can partially explore causal features from streaming features, e.g., OSFS mining PC, OCFSSF mining MB with partly spouses. The above methods provide references for mining complete causal features from streaming features.

2.2. Causal structure learning

Causal structure learning (CSL) is an important task in data mining and machine learning [38–40]. Existing CSL methods can be broadly categorized into two types: global causal structure learning (GCSL) and local causal structure learning (LCSL).

GCSL. Significant research effort has been devoted to handling GCSL. The GSBN [14] algorithm first identifies each node's MB to construct the global structure, then orients edges in a maximally consistent way using independence tests. The SLL+C [41] algorithm learns the MB of each variable for global structure and employs independence tests for orienting edges, while SLL+G [41] uses score-based methods for edge orientations. The MMHC [13] algorithm employs MMPC [42] for learning skeletons and a score function

and hill-climbing search strategy for orienting edges. GGSL [43] uses a score-based MB learning algorithm to gradually expand the local-to-whole structure. To avoid the combinatorial constraint, the NOTEARS [15] algorithm learns a global causal structure by continuous optimizing. Additionally, existing neural network methods have been demonstrated, e.g., the DAG-GNN [44] algorithm using a graph neural network, and the D-VAE [45] employing a DAG variational auto-encoder.

LCSL. As the first LCSL algorithm, LCD [46] finds causal edges by employing conditional independence relationships among every four-variable set. However, the LCD cannot identify the direct causes/effects of the target variable. Yin and Zhou et al. proposed the PCD-by-PCD [18] algorithm to discover local causal structure, which first discovers the PC of the target variable by the MMPC [13] algorithm, then sequentially discovers PCs and V-structures to orient the edges connected to the target variable. Although PCD-by-PCD can find the local causal structure of the target variable and distinguish between cause and effect, finding PCDs in large networks can become very slow. Tian et al. presented the MB-by-MB [19] algorithm, which is a sequential learning approach for using IAMB [32] to discover the MB of the target variable and start from the target variable, sequentially find MBs of features and learn local structures over MBs. Based on the concept of the PCD-by-PCD [18] algorithm, Tian et al. designed the CMB [20] algorithm that first learns the MB employing the HITON-MB [33] algorithm, then orients edges by tracking the conditional independence changes in the MB. Research shows that the efficiency of LCSL can be improved by mining more accurate MB. Yang et al. first defined the concept of the N-structure to integrate MB learning with N-structure [30]. ELCS [21] starts from the target variable by sequentially finding MBs of features to simultaneously construct local causal structures over MBs. Ling and Yu et al. designed the LCS-FS [22] algorithm to mine PC, recursively find the spouses of MBs, identify V-structures, and orient the edges connected to the target variable.

To sum up, GCSL learns a global causal structure from all variables/features in a dataset sequentially through skeleton learning and edge orientation. The global causal structure is represented by a completed partially directed acyclic graph named CPDAG. Unlike GCSL, LCSL can discover direct causal structures of a target variable from relevant variables/features. Although these methods cannot directly handle streaming features, they provide relatively complete rules for mining causal relationships between features/variables.

In summary, 1) online causal feature learning can only mine partial features in MB of the target variable, resulting in the inability to distinguish between the parents and children of the target variable; 2) existing GCSL algorithms that construct the global network in high-dimensional feature space are impractical, leading to wasted time and space and causing NP-hard problems; 3) existing LCSL algorithms cannot be applied to the dynamic feature space. Indeed, existing local causal structure learning algorithms can distinguish between the direct cause and the direct effect of a given feature. However, these algorithms frequently overlook the case of streaming features. Therefore, we propose an LCSL method, named LCSL_{SF}, for streaming features to overcome the above difficulties.

3. LCSL for streaming features

3.1. Problem setting

Here we outline the general setting of LCSL for streaming features. In the dynamic feature space, the feature space gradually increases as features continue to flow in. Let S_F be the current feature set, $S_F = \{ X_1, X_2, X_3, ..., X_i \}$, $1 \le i \le m$, where m is the size of features and X_i denotes the feature arriving at time t, and given the target variable T. Here, we use $X_i \perp T$ and $X_i \not\perp T$ to represent independence and dependence between X_i and T, respectively. If there is a directed edge connecting X_i and T, where an edge $X_i \leftarrow T$ indicates X_i is a parent (direct cause) of T, on the contrary, $X_i \rightarrow T$ indicates that X_i is a child (direct effect) of T. The focus of learning local causal structures in a dynamic feature space is to distinguish the parents and children of the target variable. The uniqueness of LCSL in streaming features, in comparison to static LCSL, presents the following challenges.

Difficulty of mining causal features by learning an approximate MB from dynamic feature space. Feature dimensionality expands over time, mainly in terms of the dynamic nature of the space, even if the feature space is infinite. When the feature space exhibits dynamics, and the feature space is not unique, it is difficult to mine the more complete MB of the target variable in real-time.

Difficulty of constructing the local causal structure from dynamic feature space. Under the dynamic feature space, the local causal structure needs to determine the relationship between the existing causal structure and the newly arriving features. Possibly, with the new feature arriving, the original cause/effect of the target variable may be changed. Therefore, discarded features may affect causal judgment. Evidently, it is also a waste of time and space to construct the local structure in real-time.

The typical notations adopted in this paper are summarized in Table 1, and briefly introduce relevant definitions and theorems as follows.

Definition 1 *(Conditional Independence, CI. [25,26,47]).* Given a set of features S, feature X is conditionally independent of feature Y, iff $P(X = x, Y = y|S) = P(X = x|S)P(Y = y|S)$, which can be denoted as $X \perp Y \mid S$.

Definition 2 *(Faithfulness. [1,31,48]).* A Bayesian network is presented by a directed acyclic graph $\mathbb{G}$ and a joint probability distribution P over a variable set S_F. $\mathbb{G}$ and P are faithful to each other iff every conditional independence present in P is entailed by $\mathbb{G}$ and the Markov condition. P is faithful iff there exists a variable $\mathbb{G}$ such that $\mathbb{G}$ is faithful to P, otherwise, P is non-faithful.

Definition 3 *(V-structure. [47]).* If there is no an edge between X and Y, and Z has two incoming edges from X and Y, respectively, then X, Z and Y form a V-structure ($X \rightarrow Z \leftarrow Y$).

Table 1
Notations and mathematical meanings.

Notations	Meanings
LCS	local causal structure
GCSL	global causal structure learning
LCSL	local causal structure learning
S_F	a feature set under the streaming features
T	a target variable. The target variable is the central node of the local causal structure, and LCSL is to explore the causes and effects of a target variable T. The T may be a class attribute or a feature of the dataset
X, Y, Z	a feature/variable
S	a set of features/variables
$X \not\perp Y \mid S$	X is conditionally dependent of Y given S
$X \perp Y \mid S$	X is conditionally independent of Y given S
P_T	parents of T
C_T	children of T
PC_T	parents and children of T, i.e., $PC_T = P_T \cup C_T$
CPC_T	a candidate set of PC_T
aMB_T	approximate Markov Blanket of T
MB_T	Markov Blanket of T
CF_{SP}	a candidate feature set for mining spouses
$Sp_T\{X\}$	a spouse of T with regard to T's child X
$CSp_T\{X\}$	a candidate $Sp_T\{X\}$
$Sep_T\{X\}$	a subset of feature that make X and T being conditional independent
$\mathbb{G}$	a direct acyclic graph

Definition 4 *(D-Separation. [3,47]).* A path L between features X and Y given conditioning set S is blocked if one of the following conditions is satisfied: 1) there is a non-collider feature within S on L; 2) there is a collider feature Z on L, whereas Z and any its descendants are not in S. Otherwise, L between features X and Y is unblocked. Here, features X and Y are D-separated given S iff each path between X and Y is blocked by S.

Definition 5 *(Markov Blanket, MB. [3,31,47]).* A Markov blanket of the target variable T (MB_T), if and only if for $\forall S_i \in S \backslash (MB_T \cup T)$, $S_i \perp T \mid MB_T$.

The MB_T consists of casual features, e.g., the parents (direct causes), children (direct effects), and spouses (other parents of the variable's children). Under the condition of non-faithfulness, mining unique MB is impractical, especially in the condition of streaming features due to the changing distributions. Therefore, we attempt to mine an approximate Markov Blanket, named aMB, to remain totally causal features in the aMB by filtering irrelevant and redundant features.

Definition 6 *(Causal Feature Selection, CFS. [1,3]).* CFS focus on identifying the MB of a variable or a subset of the MB such as parents and children (PC) without learning an entire BN structure involving all features in a dataset.

Definition 7 *(Redundant Features. [25]).* A feature X is redundant to T iff $\exists S_i \subseteq S \setminus \{X\}$, such that $P(X|S_i, T) \neq P(X|S_i)$, and has a Markov Blanket, MB(X), that is a subset of the Markov Blanket of MB(T).

Definition 8 *(Irrelevance. [25]).* A feature X is irrelevant to T iff $\forall S_i \subseteq S \setminus \{X\}$ such that $P(X|S_i, T) = P(X|S_i)$.

The relationships between features and the target variable T include strong relevance, weak relevance, and irrelevance [25]. Among them, 1) Strong relevant features belong to MB_T; 2) Irrelevant features do not belong to MB_T; 3)Weak relevant features that do not satisfy multiple independent conditions belong to MB; Otherwise, does not belong to MB_T, named redundant features.

Theorem 1 *([3,48]). In a faithful BN, for any two features* $\mathrm{X} \in \mathrm{S_F}$ *and* $\mathrm{Y} \in \mathrm{S_F}$*, if* X *and* Y *are adjacent, iff* $\mathrm{S} \subseteq \mathrm{S_F} \backslash \{X \cup Y\}$*, such that* $\mathrm{X} \perp \mathrm{Y}|\mathrm{S}$.

Theorem 2 *([3,48]). In a faithful BN, for any three features* X, Y, *and* $\mathrm{Z} \in \mathrm{S_F}$, X *and* Y *are not adjacent and* Z *is a collider (e.g.,* $X \rightarrow Z \leftarrow Y$*). If* $\mathrm{X} \perp \mathrm{T}|\mathrm{S}$, $\mathrm{S} \subseteq \mathrm{S_F} \backslash \{X \cup Y \cup Z\}$*, and* $\mathrm{X} \not\perp \mathrm{Y}| \{\mathrm{S} \cup \mathrm{Z}\}$ *then* X *is a spouse of* Y.

$X \perp Y|S$ vs. $X \not\perp Y|S$. To determine X and T are conditionally independent when S is given, iff $\rho > \alpha$, $X \perp Y|S$ holds. Otherwise, $X \not\perp Y|S$. Here, ρ is returned by G^2 test and Fisher's z test. The significance level of α represents the probability of rejecting a conditional independence hypothesis.

1) G^2 test focuses on the independence tests of discrete distribution and is an alternative to the χ^2 test, where G^2 is defined as

$$G^2 = 2 \sum_{i,j,k} S_{xyz}^{ijk} \ln \frac{S_{xyz}^{ijk} S_z^k}{S_{xz}^{ik} S_{yz}^{jk}}. \tag{1}$$

where S_{xyz}^{ijk} represents the number of features x, y, z, satisfying $x = i$, $y = j$, $z = k$ respectively. S_z^k, S_{xz}^{ik} and S_{yz}^{jk} are defined in a similar way.

2) Fisher's z test is an alternative and reliable test to assess the conditional independence tests of continuous data, where z is defined as

$$z = \frac{1}{2}\sqrt{n - |c| - 3}(\ln(\frac{1+\xi}{1-\xi})). \tag{2}$$

where n represents the number of samples/instances, c is the conditional feature, and ξ represents the overall partial relevance coefficient of the features x and y for the condition of c, the ξ is defined as

$$\xi_{(xy|z)} = \frac{\xi_{(xy)} - \xi_{(xz)}\xi_{(yz)}}{\sqrt{1-\xi_{xz}^2}\sqrt{1-\xi_{yz}^2}}. \tag{3}$$

where $\xi_{(xy)}$ represents the relevance coefficient of feature x and feature y. Assume that feature $x = (x_1, x_2, x_3 \ldots x_n)^T$ and feature $y = (y_1, y_2, y_3 \ldots y_n)^T$. Then $\xi_{(xy)}$ is defined as

$$\xi_{(xy)} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}. \tag{4}$$

where $\xi_{(xz)}$ and $\xi_{(yz)}$ and so on. If we let

$$\zeta = \frac{1}{2}\sqrt{n - |c| - 3}(\ln(\frac{1+\rho}{1-\rho})). \tag{5}$$

then asymptotically z- ζ has the standard normal distribution.

Fisher's z is a way to transform the sampling distribution of Pearson's r (i.e., the correlation coefficient) so that it becomes normally distributed [13]. The functions mentioned above hold under the null hypothesis of conditional independence between X and Y given the subset S. The null hypothesis is rejected when $\rho \leq \alpha$, and we conclude that the feature X is strongly relevant or non-redundant to Y thus far. On the contrary, the null hypothesis is accepted when $\rho > \alpha$(the significance level of $\alpha = 0.01$ or 0.05 is usually used [25]). Thus, we conclude that the feature X is redundant to Y and should be discarded.

3.2. *The* LCSL_{SF} *framework*

To address the challenge of the LCSL for streaming features, we first mine causal features from streaming features to construct aMB. Then, we use the features in aMB to explore the local causal structure of the target variable. We mine the local causal structure from streaming features, as shown in the LCSL_{SF} framework, through two key steps: online causal feature learning (Step 2) and local causal structure learning (Step 4). The following subsections present the theoretical analysis of the LCSL_{SF} framework.

Framework: The LCSL_{SF} Framework
1. Initialization
(1) The target variable T;
(2) Stream in a new feature X;
(3) The approximate Morkov Blanket $aMB_T = \varnothing$;
(4) T's parents $P_T = \varnothing$; T's children $C_T = \varnothing$;
2. Online causal feature learning via mining aMB
(5) Determine whether X and T are independent by Proposition 1(1);
(6) Determine whether $X \in PC_T$ by Proposition 2(1);
(7) If $X \notin PC_T$, determine whether X is spouse of T by Proposition 1(2) and Proposition 2(2);
(8) If $X \notin PC_T$ and $X \notin Sp_T$, then X is discarded. Otherwise, X is added to aMB_T;
3. Repeat steps (5) to (8) until no features are available
4. Local causal structure learning in aMB
(9) Construct the local causal skeleton of aMB_T by Proposition 3;
(10) Identify V-structures and orient the remaining edges by Proposition 4;
(11) Distinguish parents and children of T
5. Output P_T and C_T

3.2.1. *Online causal feature learning via mining aMB*

In real scenarios, it is difficult to obtain data with a faithfulness distribution, and the MB obtained is not unique according to Definition 2. For streaming features, we do not attempt to mine the only unrealistic MB, due to its changing and non-faithful distribution. Instead, the aMB of the target variable is mined by online irrelevant feature filtering and online redundant feature filtering.

Online irrelevant feature filtering. OL_{aMB} performs an online irrelevant feature filtering with new features flowing in, while the irrelevant features are identified and filtered by the conditional independence test.

Proposition 1. *Let X be a newly arriving feature in S_F, and Y be a feature of CPC_T. For X, Y, T, and S, the following conditions hold.*

1) *If X is independent of T in the null condition set, then $X \notin CPC_T$.*

2) *If X is added to CF_{SP}, for $\forall A, A \in CF_{SP}$, A and T are independent under the condition set that is $Sep_T\{A\} \cup \{Y\}$, i.e., $A \perp T | Sep_T\{A\} \cup \{Y\}$ holds, then $A \notin CSp_T\{Y\}$.*

Proof. For Proposition 2, 1) and 2) are proved as follows, respectively.

1) Based on Definition 1, if X is independent to T in the null condition set, then $P(X,T|\varnothing) = P(X|\varnothing)P(T|\varnothing) = P(X)P(T)$, i.e., $X \perp T \mid \varnothing$, knowing $P(X|T,\varnothing) = P(X)$ is established. Combined with Definition 8, X is an irrelevant feature, and $X \notin CPC_T$. Proposition 1(1) is proven.

2) With reference to Definitions 3 and 5, we suppose A is a parent of Y and the spouse of T. A, T, and Y will form the V-structure in a BN, and the V-structure of $A \to T \leftarrow Y$ will be satisfied. Based on Definition 4 and Theorem 2, if $A \to T \leftarrow Y$ is satisfied, makes $A \not\perp T | Sep_T\{A\} \cup Y$ and $A \perp T | Sep_T\{A\}$ hold. This conclusion contradicts the known condition that $A \perp T | Sep_T\{A\} \cup \{Y\}$. Therefore, A is not a spouse of T relative to Y, then $A \notin CSP_T\{Y\}$. Proposition 1(2) is thus proven.

Online redundant feature filtering. When a new feature is merged into a candidate feature set, a candidate feature analysis is performed to filter redundant features using a conditional independence test.

Proposition 2. *Let X be a newly arriving feature in S_F, and Y be a feature of CPC_T. For X, Y, T, and S, the following conditions hold.*

1) *If X is added to CPC_T, for $\forall B, B \in CPC_T$ and $\exists S \subseteq CPC_T \setminus \{X\}$, B and T are independent under S, i.e., $B \perp T | S$ holds, then $B \notin PC_T$.*

2) *If X is added to $CSp_T\{Y\}$, for $\forall C$, $C \in CSp_T\{Y\}$ and $\exists S \subseteq CPC_T \cup CSP_T\{Y\} \setminus \{C\}$, C and T are independent under S, i.e., $C \perp T | S \cup \{Y\}$ holds, then $C \notin Sp_T\{Y\}$.*

Proof. Suppose $B \in PC_T$, invoking Definition 5, B is the parent or child of T. Therefore, B denotes strong relevance to T. Combined with Definition 7, $\exists S \subseteq CPC_T \setminus \{X\}$ indicates that $B \not\perp T | S$ holds. This conclusion contradicts the known condition that $B \perp T | S$, and it can be shown that B is a redundancy feature. Therefore, $B \notin PC_T$. Proposition 2(1) is proven. Referring the same logic, to prove $C \notin Sp_T\{Y\}$, Proposition 2(2) is proven.

Proposition 3. *If X is an irrelevant or redundant feature, then $X \notin MB_T$.*

Proof. The proof of Proposition 3 is based on the following fact. For known conditions, if X is an irrelevant feature, invoking Definition 8, X is independent of T, then $X \notin MB_T$. If X is a redundant feature, invoking Definition 7, X is weakly related with T, and MB_X is a subset of MB_T. Combined with Theorem 2, MB_X and X are adjacent, MB_X and T are adjacent, combined with Proposition 2, X is independent of T under multiple conditions, X is neither PC nor a spouse of T, then $X \notin MB_T$.

Proposition 3 proves that filtered irrelevant and redundant features are not causal. That is, the causal features are retained in the aMB. Specifically, OL_{aMB} is an approximate MB that identifies the target variable in an online manner. With the arrival of new features, irrelevant features are directly abandoned and relevant features are incorporated into aMB. After the filtering of online redundant features between features within the aMB, the causal features in the original aMB may become redundant and be discarded, and the new causal features remain in the aMB. In other words, causal features in aMB are strongly relevant or non-redundant with the target variable/feature.

3.2.2. Local causal structure learning in aMB

Existing LCSL algorithms based on static feature space function are divided into three steps, namely, first finding the PC of the target variable, then determining the local causal skeleton by iteratively finding the PC of the PC, and finally differentiating the direction. However, in non-faithful BN, the MB found is not unique, and as the feature space increases, the iterative search for MB is time-consuming. Therefore, the aMB of the target variable can be obtained in real-time by processing in the first step. The next step is to learn the local causal structure of the target variable on the aMB, and learning the local causal structure is divided into constructing the local causal skeleton and orienting edge.

Constructing the local causal skeleton. To construct the local causal skeleton, the aMB of the target variable is obtained by multi-layer filtering of the dynamic feature space, and a batch process is performed at the aMB.

Proposition 4. *For feature $X \in S_F$, and X is the feature of PC_T. If $\forall X$, $X \in PC_T$, X and Y form a skeleton (undirected edge), i.e., $X - T$, then there are edges between the pair of nodes of X and Y.*

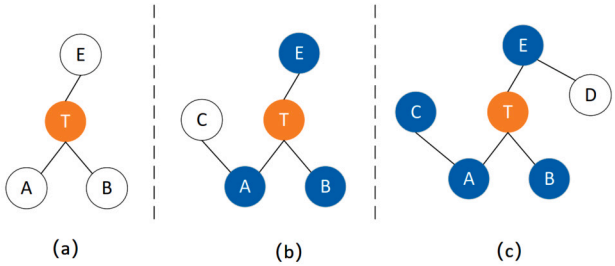


Fig. 3. An example of constructing the approximate Markov Blanket skeleton.

Proof. The proof of Proposition 4 is based on the following fact. For known conditions, $X \in PC_T$ holds. Invoking Definition 5, X is the parent or child of T. On the contrary, T is the parent or child of X, i.e., $T \in PC_X$. Combined with Theorem 1, the nodes of PC that obtain T is always adjacent to T. Therefore, there are edges between the pair of X and T. Proposition 4 is thus proven.

We propose that Proposition 4 is used to construct the skeleton of the aMB, and use Fig. 3 to validate Proposition 4. Suppose $aMB_T = A, B, C, D, E, F$. According to Proposition 4, there exists an edge between T and the PC of T, as in Fig. 3(a). Then, find the PC of each feature X in PC_T, such as there exists an edge between A and C, as in Fig. 3(b), and there exists an edge between E and D, as in Fig. 3(c).

Orienting edges. To more accurately distinguish the orientation of the local skeleton, we execute the conditional independence test to determine the V-structure, and for the orientation of undirected edges, use the Meek rules to update the orientation of undirected edges according to the directed V-structure.

Proposition 5. *Suppose any three features X, Y, and $Z \in S_F$ that form a local skeleton, such as $X - Y - Z$. If X and Z are independent under the condition set that is $Sep_X\{Z\}$, and X and Z are not independent under the condition set that is $Sep_X\{Z\} \cup \{Y\}$, i.e., $X \perp Z \mid Sep_X\{Z\}$ and $X \not\perp Z | Sep_X\{Z\} \cup \{Y\}$, then $X \rightarrow Y \leftarrow Z$.*

Proof. The proof of Proposition 5 follows from the following fact. For known conditions $X - Y - Z$ holds, invoking Proposition 4, X and Z are the PC of Y. Combined with Theorem 1, obtaining the following conditions $X \not\perp Y$ and $Y \not\perp Z$. For $X \not\perp Y$ and $X \perp Z$, indicate there is no edges between the pair of nodes of X and Z. Based on Definition 4 and Theorem 2, if $X \perp Z \mid Sep_X\{Z\}$ and $X \not\perp Z | Sep_X\{Z\} \cup \{Y\}$ hold, then the condition is satisfied by the V-structure $A \rightarrow B \leftarrow C$. Thus, under this structure, X is the spouse of Z relative to Y.

Based on the theoretical support of the above propositions, the proposed the LCSL_{SF} framework can reasonably construct the causal structure of the target variable for streaming features through two steps, i.e., finding the aMB, and constructing the causal structure between the causal features in aMB and the target variable.

4. Algorithms for LCSL for streaming features

4.1. The OL_{aMB} algorithm for online LCSL via mining aMB

According to Step 2 in the LCSL_{SF} framework and the theoretical analysis of Section 3.2.1, we present the main ideas of the subroutine and describe the OL_{aMB} algorithm step by step. OL_{aMB} performs the phase of finding the aMB in two steps: determining the relevance between X and T and learning aMB_T of T.

Step 1: For the newly arriving feature X, OL_{aMB} first determines whether X is non-conditionally independent of the target variable T. If $X \perp T \mid \emptyset$ holds (line 6), X is irrelevant to T and adds X to the set of CF_{SP} (line 7). Otherwise, X is added to the set of CPC_T (line 9), and changes *stop* to 1 (line 10).

Step 2: Filter non-PC from the set of CPC_T and filter non-spouse from the set of CSp_T.

1) Filter non-PC from CPC_T: When $stop = 1$, and if $\exists S \subseteq CPC_T \setminus \{X\}$, such that $X \perp T \mid S$ holds (line 14), X is the redundant feature that is removed from CPC_T (line 15), and adds X into CF_{SP} to update the set containing all the features independent of T (line 16). Meanwhile, $Sep_T\{X\}$ maintains the set S(line 17), and $CSp_T\{X\}$ must be empty (line 18). Otherwise, add X into aMB_T (line 20).

2) Filter non-spouse from CSp_T: If CPC_T and CF_{SP} are not empty (line 21), OL_{aMB} learns possible spouses of T from CF_{SP}. For each feature A in CPC_T and each feature B in CF_{SP}, if $B \not\perp T \mid Sep_T\{B\} \cup \{A\}$ holds (line 24), B may be the spouse of T with regard to T's child A, add B into $CSp_T\{A\}$ (line 25), and update aMB_T (line 26). As B is added into $CSp_T\{A\}$, and $CSp_T\{A\}$ is not empty, for each feature C in $CSp_T\{A\}$ (line 27), if $\exists S \subseteq CPC_T \cup CSp_T\{A\} \setminus \{C\}$, such that $C \perp T \mid S \cup \{A\}$ holds (line 29), C is not a spouse of T, and C must be removed from aMB_T (line 30).

Algorithm 1: The OL_{aMB} Algorithm — cost.

Input: T:target variable, X:The arriving feature;
Output: aMB_T: approximate MB of T;

1 **begin**
2 Initialize: $aMB_T \leftarrow \emptyset$; $CPC_T \leftarrow \emptyset$; $CSp_T \leftarrow \emptyset$; $CF_{SP} \leftarrow \emptyset$; $stop = 0$;
3 Repeat
4 Receive a new feature X; — $O(1)$
5 /* ***Step 1: Determine the relevance between X and T*** */
6 **if** $X \perp T \mid \emptyset$ **then**
7 $CF_{SP} \leftarrow CF_{SP} \cup \{X\}$; — $O(1)$
8 **else**
9 $CPC_T \leftarrow CPC_T \cup \{X\}$;
10 $stop = 1$;
11 /* ***Step 2: Learn aMB***$_T$ */
12 **if** *(stop)* **then**
13 **for** $X \in CPC_T$ **do**
14 **if** $X \perp T \mid S$ *for* $S \in CPC_T \setminus \{X\}$ **then**
15 $CPC_T \leftarrow CPC_T \setminus \{X\}$; — $O(|CPC_T|2^{|CPC_T|})$
16 $CF_{SP} \leftarrow CF_{SP} \cup \{X\}$;
17 $Sep_T\{X\} \leftarrow S$;
18 $CSp_T\{X\} \leftarrow \emptyset$;
19 **else**
20 $aMB_T \leftarrow aMB_T \cup \{X\}$;
21 **if** *nonempty* (CPC_T) && *nonempty*(CF_{SP}) **then**
22 **for** *each* $A \in CPC_T$ **do**
23 **for** *each* $B \in CF_{SP}$ **do**
24 **if** $B \not\perp T \mid Sep_T\{B\} \cup \{A\}$ **then**
25 $CSp_T\{A\} \leftarrow CSp_T\{A\} \cup \{B\}$; — $O(|CSp_{T\{X\}}|2^{|CPC_T|+|CSp_T\{X\}|})$
26 $aMB_T \leftarrow aMB_T \cup \{B\}$;
27 **for** *each C in nonempty* $CSp_T\{A\}$ **do**
28 $S \leftarrow CPC_T \cup CSp_T\{A\} \setminus \{C\}$;
29 **if** $C \perp T \mid S \cup \{A\}$ **then**
30 $aMB_T \leftarrow aMB_T \setminus \{C\}$;
31 Until no available features to arrive;
32 **return** aMB_T

4.2. *The* $LCSL_{SF}$ *algorithm for LCSL in aMB*

According to the $LCSL_{SF}$ framework and the theoretical analysis in Section 3.2.2, we present the $LCSL_{SF}$ algorithm. First, we determine the skeleton of the aMB_T, then identify the orientation by V-structure and update the remaining edge orientation using the Meek rules, and finally, distinguish between parent and child nodes.

Step 1: First, the dynamic feature space is filtered in real-time using OL_{aMB} to obtain the aMB_T (line 2), and add T to aMB_T (line 4). $LCSL_{SF}$ defines an adjacency matrix G (line 4), which is stored the correspondence between features, wherein if $a \rightarrow b$, denoted as $G(a, b) = 1$, and if there are no edges between features a and b, denoted as $G(a, b) = 0$. G is denoted by an $n * n$ matrix (n denotes the number of features of the mined aMB_T). Then, sequentially learn the PC of the target Y in the aMB_T feature space with the extant HITON-PC algorithm (line 7), where any other well-established PC mining method can be used. Meanwhile, construct the skeleton between Y and PC_Y to obtain a local skeleton (line 9).

Step 2: $LCSL_{SF}$ distinguishes between the parents and children of T. The variables A, B, and C denote the nodes of aMB_T, PC of A, and PC of B, respectively. This indicates that there are edges between A and B, and edges between B and C, i.e., $A - B - C$ (lines 11-13). Next, to construct more and more accurate V-structures, we compare the size of PC_A and PC_C. If the size of PC_A is greater than or equal to the size of PC_C (line 14), the separation set of PC_A is used as a condition, i.e., $Sep_A\{C\} \subseteq PC_A$, and if $A \perp C \mid Sep_A\{C\}$ and $A \not\perp C \mid Sep_A\{C\} \cup B$ are satisfied, orienting V-structure $A \rightarrow B \leftarrow C$ (lines 16-17). Otherwise, the separation set of PC_C is used as a condition, i.e., $Sep_C\{A\} \subseteq PC_C$, and if $C \perp A \mid Sep_C\{A\}$ and $C \not\perp A \mid Sep_C\{A\} \cup B$ are satisfied, the V-structure $C \rightarrow B \leftarrow A$ is oriented (lines 20-21). Then using the Meek rules, other edges in G are oriented according to the oriented V-structures (line 22). After the orientations in Step 2, the parents and children of T are distinguished. Then outputs parents and children of T (line 24).

In summary, with the assumption that all independence tests are reliable, OL_{aMB} can acquire the aMB_T of the target variable by removing the non-PC and non-spouse, which is a pre-processing step for learning the local causal structure of the streaming feature. Based on the above assumptions and causal features in aMB mined by OL_{aMB}, $LCSL_{SF}$ can obtain the direct causes/effects of the target variable by batch processing the aMB_T.

Algorithm 2: The LCSL_{SF} Algorithm — cost.

Input: T:target variable, X:The arriving feature;
Output: $[P_T, C_T]$: local causal structure of T;
begin
  $aMB_T = \text{OL}_{\text{aMB}}(T, X)$;
  $aMB_T = aMB_T \cup T$;
  $G = zeros(|n||n|)$; — $O(1)$
  /* ***Step 1: Construct the skeleton*** */
  for *each Y in nonempty* aMB_T **do**
    PC_Y = GetPC *(Y)*;
    for *each Z in* PC_Y **do**
      $G(Z, Y) = G(Y, Z)$; //Constructing the skeleton between Y and Z — $O(|PC|2^{|PC|})$
  /*$\ast$***Step 2: Identify the orientation**$\ast$*/
  for *each A in* aMB_T **do**
    for *each B in* PC_A **do**
      for ***for*** *each C in* PC_B **do**
        if $|PC_A| \geq |PC_C|$ **then**
          if $A \perp C \mid Sep_A\{C\}$ && $A \not\perp C \mid Sep_A\{C\} \cup B$ **then**
            $G(A, B)$ = edge; //Adding edge: A → B — $O(|PC|2^{|PC|})$
            $G(C, B)$ = edge; //Adding edge: C → B
        else
          if $A \perp C \mid Sep_C\{A\}$ && $A \not\perp C \mid Sep_C\{A\} \cup B$ **then**
            $G(A, B)$ = edge; //Adding edge: A → B
            $G(C, B)$ = edge; //Adding edge: C → B
    Orienting other edges in G by using Meek rules;
  Save the parents, children, and spouses of T in G to P_T, C_T and S_T
  return $[P_T, C_T]$ //Outputing parents and children of T as causes and effects, respectively

4.3. Computational complexity

The complexity of OL_{aMB} is mainly cost by removing non-spouse and non-PC in the phase of learning aMB_T. Suppose $|M|$ is the number of features arriving at the current stage. For OL_{aMB}, the computational cost takes $O(|\boldsymbol{CPC}_T|2^{|\boldsymbol{CPC}_T|})$ on CI tests to remove non-PC nodes and $O(|\boldsymbol{CSp}_T|2^{|\boldsymbol{CPC}_T|+|\boldsymbol{CSp}_T|})$ to remove non-spouse nodes. In the best case, the time complexity is $O(|\boldsymbol{M}|)$ when all incoming features are independent of the target variable, i.e., no PC and spouse throughout the process. However, in the worst case, the complexity is $O(|\boldsymbol{M}| + |\boldsymbol{M}||\boldsymbol{CPC}_T|2^{|\boldsymbol{CPC}_T|} + |\boldsymbol{CPC}_T||\boldsymbol{M} - \boldsymbol{CPC}_T||\ \boldsymbol{CSP}_T|2^{|\boldsymbol{CPC}_T|+|\boldsymbol{CSp}_T|}) = O(|\boldsymbol{M}||\boldsymbol{CPC}_T|2^{|\boldsymbol{CPC}_T|+}|\boldsymbol{CPC}_T||\boldsymbol{M} - \boldsymbol{CPC}_T||\boldsymbol{CSp}_T|2^{|\boldsymbol{CPC}_T|+|\boldsymbol{CSp}_T|})$, i.e., all incoming features are MB of the target variable. Naturally, the above best/worse situation is almost impossible.

For LCSL_{SF}, by first invoking the OL_{aMB} algorithm, the time complexity of algorithm LCSL_{SF} is affected by the output of OL_{aMB}, and we assume that the output of OL_{aMB} is $|N|$. Based on the feature space of the aMB_T, the skeleton is first constructed with the time complexity of $O(|\boldsymbol{PC}|2^{|\boldsymbol{PC}|})$, and then the orientation is determined with the time complexity of $O|\boldsymbol{PC}|2^{|\boldsymbol{PC}|})$. In the best case, all input features are independent of T, i.e., $N = 0$, and there is no need to construct for local causal structures of the target variable. Therefore, the best time complexity of LCSL_{SF} is $O(|M|)$. In the worst case, all incoming features are MB of the target variable, OL_{aMB} output is M, i.e., $N = M$, and LCSL_{SF} must learn a whole structure. Learning the local causal structure with all features with time complexity $O(|\boldsymbol{M}||\boldsymbol{PC}|2^{|\boldsymbol{PC}|})$, and finally determining the direction with time complexity $O(|\boldsymbol{M}||\boldsymbol{PC}|^2 2^{|\boldsymbol{PC}|})$. Therefore, the worst time complexity of LCSL_{SF} is $O(|\boldsymbol{M}||\boldsymbol{PC}|2^{|\boldsymbol{PC}|} + |\boldsymbol{M}||\boldsymbol{PC}|^2 2^{|\boldsymbol{PC}|}) = O(|\boldsymbol{M}||\boldsymbol{PC}|^2 2^{|\boldsymbol{PC}|})$.

5. Experimental studies

In this section, we mainly focus on the performance of the algorithms OL_{aMB} and LCSL_{SF} in mining causal features, and constructing the local causal structure, respectively. In detail, we aim to answer the following research questions (RQs):

- *RQ1*: Can OL_{aMB} mine aMB with more complete streaming causal features than other state-of-the-art online algorithms?
- *RQ2*: Does LCSL_{SF} outperform its rivals in dealing with static feature space to generate local causal structure?
- *RQ3*: Does LCSL_{SF} outperform its rivals in processing dynamic feature space to generate local causal structure?

5.1. Experimental setting

Datasets. The datasets include two parts: synthetic and real, as shown in Table 2. These datasets satisfy two key attributes: streaming and casual through the following settings.

Table 2
Statistics of the datasets in experiments. The symbol "| • |" indicates the length of the PC set.

RQs	Category	Datasets	Features	Instances	Datasets	Features	Instances
RQ1	real	sylva	216	14374	lung	3313	13086
		LSVT	311	126	prostate	6034	102
		madelon	500	2000	leukemia	7071	72
		marti1	1024	500			

RQs	Category	Datasets	Features	Instances	Num.Edges	Max In/Out Degree	Min/Max |PCset|	Domain Range
RQ1 & RQ2 & RQ3	synthetic	Win95pts	76	500/5000	112	7/10	1/10	2-6
		Child5	100	500/5000	126	2/7	1/8	2-6
		Insurance5	135	500/5000	284	5/8	1/10	2-5
		Alarm5	185	500/5000	265	4/6	1/8	2-4
		Child10	200	500/5000	257	2/7	1/8	2-6
		Insurance10	270	500/5000	556	5/8	1/11	2-5
		Alarm10	370	500/5000	570	4/7	1/9	2-4
RQ2 & RQ3	synthetic	Child	20	500/5000	25	2/7	1/8	2-6
		Insurance	27	500/5000	52	3/7	1/9	2-5
		Alarm	37	500/5000	46	4/5	1/6	2-4
		Hepar2	70	500/5000	123	6/17	1/19	2-4
		Andes	223	500/5000	338	1/2	0/12	3-3
		Link	724	500/5000	1125	3/14	0/17	2-4

- *Streaming setting*. Following general streaming feature research [25–27], we receive features one by one from synthetic/real datasets to simulate the streaming pattern. As features continue to arrive in an original/random sequence, irrelevant and redundant features are filtered in real-time under the unknown feature space and fixed instances in a dataset.
- *Causal setting*. We select typical and widespread synthetic datasets with causal attributes to certify the efficiency of discovering causal features and structures, e.g., benchmark BNs. Meanwhile, for fairness in analyzing the performances of mining causal features, we also utilize some representative real datasets with causal/non-causal attributes, e.g., Causality Workbench [49] on our $\mathrm{OL}_{\mathrm{aMB}}$ and its rivals. [1,25–27].

Table 2 show thirteen synthetic and seven real datasets that we selected. Among them, seven synthetic and seven real datasets for RQ1, and thirteen synthetic datasets for RQ2 and RQ3. For synthetic datasets, we utilize thirteen standard benchmark BNs.[1] Each BNs dataset consists of 10 datasets with large/small-sized instances (5000/500 per dataset), respectively. The real datasets come from the benchmark datasets of some causal/non-causal FS algorithms, such as OSFS, SAOLA, and OCFSSF. Herein, the madelon is from the Neural Information Processing System (NIPS) 2003 Feature Selection Challenge [50], and the lung and marti1 are datasets from the Causality Workbench [49], and the prostate and leukemia datasets are frequently studied public microarray datasets.

Evaluation Metrics.

- *Accuracy*: The number of accurately predicted data samples divided by the total number of data samples in a dataset. We report the prediction accuracy on nine classifiers in MATLAB's classifier box, i.e., Decision Tree (Fine Tree, Medium Tree, and Coarse Tree), KNN (Fine KNN, Medium KNN, and Weighted KNN) and SVM (Linear SVM, Quadratic SVM, and Medium Gaussian SVM), to evaluate selected features. To avoid bias in error estimation, we used ten-fold cross-validation for all datasets.
- *SHD*: The number of total error edges, which contains undirected edges, reverse edges, missing edges and extra edges. The smaller value of *SHD* is better.
- *Precision*: The number of correctly predicted arrowheads in the output divided by the number of edges in the output of an algorithm (i.e., the variables in the output belonging to the true parents and children).
- *Recall*: The number of correctly predicted arrowheads in the output divided by the number of true arrowheads in the true local BN structure (i.e., the number of parents and children of a target variable).
- $F1$: The $F1$ score is the harmonic average of the Precision and Recall, where $F1 = 1$ is the best case (perfect Precision and Recall) while $F1 = 0$ is the worst case. $F1 = 2 \times (Precision \times Recall)/(Precision + Recall)$.

Statistical Test. To analyze the statistical performance of algorithms on each evaluation metric, Friedman's and Nemenyi's tests are used to calculate the average ranks and critical difference, respectively. The Friedman's statistic is expressed as follows:

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} \tag{6}$$

where

[1] Synthetic datasets can be found in https://pages.mtu.edu/~lebrown/supplements/mmhc_paper/mmhc_index.html.

Table 3
Detailed descriptions of OL_{aMB} and $LCSL_{SF}$ and compared algorithms, respectively. The symbol "–" indicates that the algorithm can partially mine causal features.

Functions	Methods	Causal	Streaming	Description
FS	OSFS	–	✓	Only learn the PC of the target variable.
	SAOLA	✗	✓	The set of selected features contains partially redundant features.
	OCFSSF	✓	✓	Learning the MB of the target variable, which distinguishes between PC and partially spouse.
	OL_{aMB}	✓	✓	Irrelevant and redundant features are removed to retain causal features.
LCSL	CMB	✓	✗	Learn MB using the HITON-MB algorithm, and orient edges by conditional independence.
	PCD-by-PCD	✓	✗	Recursively mien PCs by MMPC algorithm and find V-structures using separating sets.
	MB-by-MB	✓	✗	Learn MB of T, i.e., MB_T, and MB of the elements of MB_T by a sequential learning method.
	ELCS	✓	✗	Learn MB using the EEMB algorithm to distinguish between PC and spouse in real time.
	LCS-FS	✓	✗	Learn PC through feature selection, and find separation set from PC for identifying V-structure.
	$LCSL_{SF}$	✓	✓	Construct the local causal structure of T in real time based on the aMB mined by OL_{aMB} and learned V-structure.

$$\chi_F^2 = \frac{12N}{k(k+1)}\left(\sum_{i=1}^{k} R_i^2 - \frac{k(k+1)^2}{4}\right) \tag{7}$$

where N and k are the numbers of datasets and methods, respectively; R_i $(i = 1, 2, \cdots, k)$ represents the mean rank of the i-th methods on all datasets. We conduct the Friedman test at the 5% significance level, and the null hypothesis (i.e., all algorithms are equivalent) is rejected.

Once the null hypothesis of the Friedman test is rejected, we proceed with the Nemenyi test as a post hoc test. In the Nemenyi test, the significant differences between algorithms are reflected by average ranks differing by CD_α, as shown in the following formula:

$$CD_\alpha = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \tag{8}$$

where q_α denotes the critical tabulated value. If the average rank of the compared algorithm and $LCSL_{SF}$ is within one CD_α, they are not significantly different.

Baselines. To validate the performances, we evaluate our OL_{aMB} and $LCSL_{SF}$ algorithms separately, comparing OL_{aMB} with state-of-the-art online feature selection algorithms, i.e., OSFS, SAOLA, and OCFSSF. Meanwhile, $LCSL_{SF}$ is compared with five state-of-the-art LCSL algorithms, i.e., with CMB, PCD-by-PCD, MB-by-MB, ELCS, and LCS-FS on synthetic datasets. Table 3 shows brief descriptions of OL_{aMB}, $LCSL_{SF}$ and compared algorithms.

Implementation Details. All experiments are carried out on Ubuntu 20.04 and MATLAB 2017 with Intel(R) i7-117000K, 3.6 GHz CPU, and 32 GB of RAM. The OSFS and SAOLA algorithms are implemented by LOSF library[2] in MATLAB, and the source code for the OCFSSF algorithm is available from the authors. For the five static local structure learning algorithms, the CMB, PCD-by-PCD, and MB-by-MB algorithms are implemented by the Causal Learner library[3] in MATLAB, while the source code for the LCS-FS and ELCS algorithms are provided by the authors. In the experiments, the conditional independence tests of G^2/ Fisher's z are used for discrete/ continuous data, respectively. The statistical significance level is set to 0.01.

5.2. Comparison with online algorithms (RQ1)

To evaluate the performance of OL_{aMB} in mining causal features, the comparison experiment is conducted with OSFS, SAOLA, and OCFSSF algorithms in terms of the prediction accuracy on the nine classifiers with synthetic and real datasets, respectively. The prediction accuracy of the OL_{aMB} and its rivals with respect to the synthetic datasets is shown in Table 4. On the Decision Tree, KNN, and SVM classifiers, the average accuracy of OL_{aMB} is significantly higher than that of OSFS and SAOLA, which is 2.65% and 3.43%, respectively, and slightly higher than OCFSSF. This is because of the low number of features, which filters out useful features.

For prediction accuracy of the real datasets is shown in Table 5. OL_{aMB} achieves the highest accuracy on these classifiers. For the classifiers, e.g., Decision Tree, KNN, and SVM, the average accuracy of OL_{aMB} is higher than that of OSFS, SAOLA, and OCFSSF (3.92%, 2.52%, and 1.12% higher, respectively). Meanwhile, Fig. 4 compares the average classification accuracy of each dataset on Decision Tree, KNN, and SVM classifiers. We observed that the classification performance of OL_{aMB} under each classifier is superior to that of OSFS, SAOLA, and OCFSSF. Therefore, OL_{aMB} exhibits higher prediction accuracy than OSFS, SAOLA, and OCFSSF.

5.3. Comparison with LCSL algorithms for static feature space(RQ2)

In the experiments, we evaluate $LCSL_{SF}$ against five static LCSL algorithms, i.e., CMB, PCD-by-PCD, MB-by-MB, ELCS, and LCS-FS on synthetic datasets, and use the metrics of SHD, Precision, Recall, and F1 to evaluate their performance. We run these algorithms ten times on each benchmark dataset with sample sizes of 500 and 5000, respectively. For the Link dataset, due to their high number

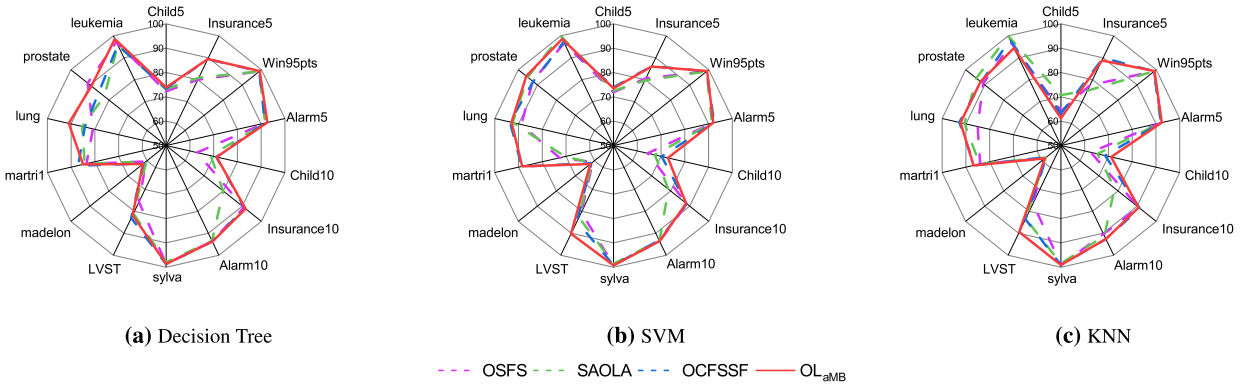
[2] The LOSF library is available at https://github.com/kuiy/LOFS.
[3] The Causal Learner library is available at https://github.com/kuiy/CausalLearner.

Table 4
Prediction accuracy of OL_{aMB} and its rivals on synthetic datasets (%).

Algorithms	Average accuracy for classifiers on synthetic datasets (%)								
OSFS	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	84.44	84.07	84.59	83.84	84.61	84.33	78.21	84.24	84.56
	Decision Tree average: 84.37			SVM average: 84.26			KNN average:82.34		
SAOLA	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	83.81	83.73	83.99	82.36	83.79	83.30	79.61	82.87	82.41
	Decision Tree average:83.84			SVM average:83.15			KNN average: 81.63		
OCFSSF	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	87.84	88.09	85.24	85.07	85.96	**87.49**	83.33	**85.73**	**86.70**
	Decision Tree average:87.06			SVM average:86.17			KNN average: **85.25**		
OL_{aMB}	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	88.07	**88.50**	**85.64**	**85.56**	**86.84**	86.70	**84.10**	85.10	86.30
	Decision Tree average: **87.40**			SVM average: **86.37**			KNN average: 85.17		

Table 5
Prediction accuracy of OL_{aMB} and its rivals on real datasets (%).

Algorithms	Average accuracy for classifiers on real datasets (%)								
OSFS	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	83.04	84.36	84.4	87.13	79.84	86.4	80.75	85.93	84.07
	Decision Tree average: 83.94			SVM average:84.45			KNN average: 85.59		
SAOLA	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	83.2	84.19	84.27	88.57	80.67	88.14	84.83	88.06	86.6
	Decision Tree average: 83.89			SVM average:85.8			KNN average: 86.5		
OCFSSF	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	84.76	85.06	85.64	88.04	88.63	87.79	85.6	**87.79**	87.81
	Decision Tree average: 85.15			SVM average:88.15			KNN average: 87.07		
OL_{aMB}	Fine Tree	Medium Tree	Coarse Tree	Liner SVM	Quadratic SVM	Medium Gaussian SVM	Fine KNN	Medium KNN	Weighted KNN
	87.29	**86.31**	**86.24**	**90.54**	**90.06**	**88.31**	**86.49**	**87.79**	**88.19**
	Decision Tree average: **86.61**			SVM average: **89.64**			KNN average: **87.49**		



(a) Decision Tree **(b)** SVM **(c)** KNN

Fig. 4. Average prediction accuracy on Decision Tree, SVM, and KNN classifiers.

of features and computer configuration problems, we randomly select approximately 10% of the features in each BN to learn the local causal structure.

Performance evaluation. Tables 6-9 report the experimental results of the average SHD, Precision, Recall, and F1 for synthetic datasets, and combined with Fig. 5, the following observations are obtained through analysis.

Table 6
SHD(↓) results (Mean SHD ± Standard deviation) of $LCSL_{SF}$ and its rivals on static feature space with 500/5000 instances.

	500-instances						5000-instances					
Datasets	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$
Child	1.53 ± 0.24	2.01 ± 0.16	1.91 ± 0.45	**1.48 ± 0.22**	1.83 ± 0.45	1.68 ± 0.16	0.85 ± 0.11	0.95 ± 0.07	1.80 ± 0.30	1.08 ± 0.32	2.08 ± 0.10	**0.78 ± 0.09**
Insurances	3.07 ± 0.14	3.01 ± 0.09	3.31 ± 0.19	3.00 ± 0.16	2.92 ± 0.19	**2.79 ± 0.15**	2.27 ± 0.25	1.91 ± 0.15	2.56 ± 0.27	1.64 ± 0.14	3.14 ± 0.29	**1.40 ± 0.08**
Alarm	1.37 ± 0.13	1.35 ± 0.08	1.54 ± 0.20	1.35 ± 0.10	1.48 ± 0.19	**1.16 ± 0.10**	0.85 ± 0.13	0.46 ± 0.05	2.09 ± 0.13	0.43 ± 0.06	1.08 ± 0.08	**0.41 ± 0.09**
Child5	1.75 ± 0.22	2.26 ± 0.10	2.58 ± 0.12	1.99 ± 0.06	**1.41 ± 0.09**	1.98 ± 0.13	0.95 ± 0.08	0.89 ± 0.08	3.95 ± 0.13	0.78 ± 0.24	1.21 ± 0.05	**0.77 ± 0.06**
Insurances5	3.29 ± 0.06	3.21 ± 0.08	3.69 ± 0.10	3.18 ± 0.07	3.01 ± 0.08	**2.58 ± 0.08**	2.39 ± 0.06	2.00 ± 0.06	4.46 ± 0.11	1.64 ± 0.10	2.80 ± 0.03	**1.28 ± 0.06**
Alarm5	2.11 ± 0.07	1.94 ± 0.04	2.48 ± 0.10	1.98 ± 0.07	1.81 ± 0.09	**1.90 ± 0.06**	1.45 ± 0.04	1.12 ± 0.05	4.64 ± 0.17	1.11 ± 0.04	1.51 ± 0.05	**0.97 ± 0.05**
Child10	1.84 ± 0.10	2.30 ± 0.06	2.96 ± 0.05	2.09 ± 0.05	**1.42 ± 0.07**	2.04 ± 0.08	0.99 ± 0.04	0.94 ± 0.05	4.54 ± 0.10	**0.72 ± 0.20**	1.26 ± 0.11	0.76 ± 0.04
Insurances10	3.28 ± 0.05	3.18 ± 0.06	3.83 ± 0.06	3.16 ± 0.05	2.97 ± 0.07	**2.70 ± 0.06**	2.35 ± 0.08	1.89 ± 0.04	4.89 ± 0.09	1.75 ± 0.11	2.74 ± 0.02	**1.31 ± 0.04**
Alarm10	2.39 ± 0.04	**2.18 ± 0.06**	2.97 ± 0.08	2.23 ± 0.04	2.19 ± 0.07	2.19 ± 0.05	1.60 ± 0.04	1.37 ± 0.03	5.78 ± 0.05	1.23 ± 0.06	1.71 ± 0.03	**1.13 ± 0.04**
Hepar2	3.38 ± 0.16	3.41 ± 0.08	3.68 ± 0.19	3.31 ± 0.19	3.25 ± 0.16	**3.11 ± 0.15**	2.23 ± 0.08	2.68 ± 0.04	4.42 ± 0.34	2.02 ± 0.09	3.24 ± 0.11	**1.94 ± 0.07**
win95pts	2.59 ± 0.11	2.72 ± 0.07	2.77 ± 0.14	2.38 ± 0.11	2.54 ± 0.09	**2.11 ± 0.07**	1.95 ± 0.06	1.78 ± 0.09	3.36 ± 0.16	1.43 ± 0.06	2.08 ± 0.06	**1.28 ± 0.06**
Andes	2.20 ± 0.03	2.12 ± 0.40	4.52 ± 0.11	**1.78 ± 0.00**	2.15 ± 0.04	**1.78 ± 0.00**	1.57 ± 0.06	1.32 ± 0.04	9.06 ± 0.19	0.97 ± 0.00	2.15 ± 0.05	**0.94 ± 0.00**
Link	4.32 ± 0.53	4.42 ± 0.58	4.12 ± 0.31	4.25 ± 0.43	4.37 ± 0.48	**4.00 ± 0.39**	2.62 ± 0.32	2.22 ± 0.18	4.28 ± 0.32	2.60 ± 0.31	3.95 ± 0.46	**2.05 ± 0.18**
w/t/l	10/0/3	12/0/1	13/0/0	11/1/1	10/0/3	——	13/0/0	13/0/0	13/0/0	12/0/1	13/0/0	——

Table 7
Precision (↑) results (Mean Precision ± Standard deviation) of $LCSL_{SF}$ and its rivals on static feature space with 500/5000 instances.

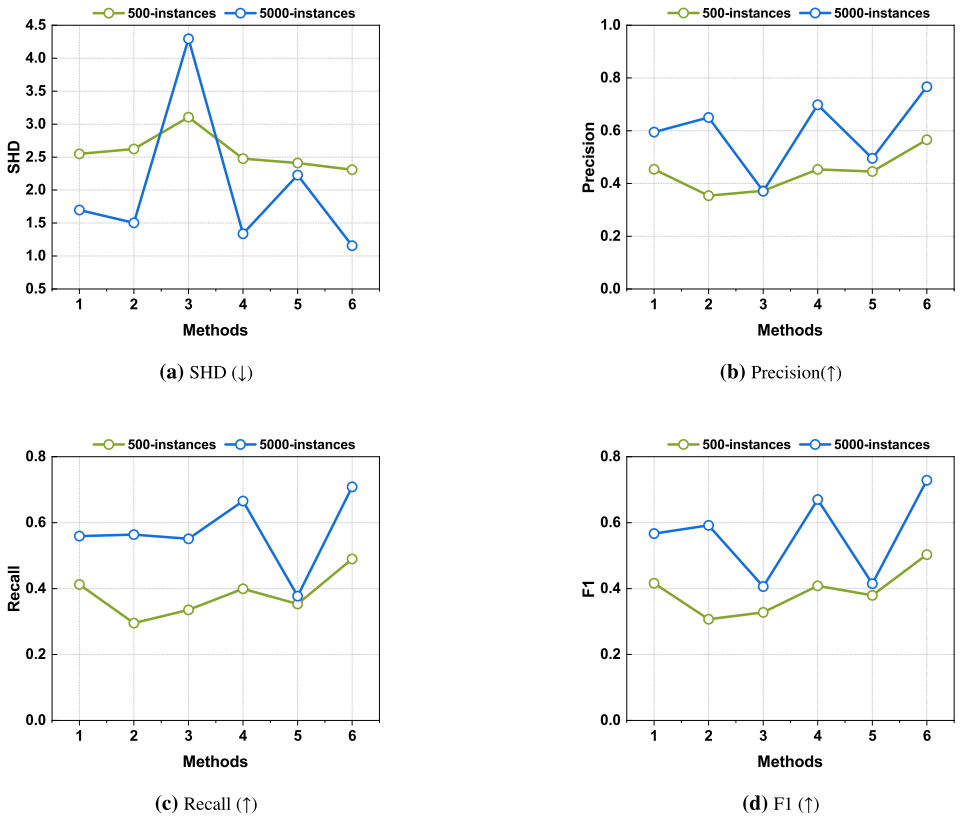
	500-instances						5000-instances					
Datasets	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$
Child	**0.59 ± 0.10**	0.30 ± 0.08	0.46 ± 0.26	0.54 ± 0.10	0.32 ± 0.22	0.58 ± 0.06	0.68 ± 0.04	0.63 ± 0.02	0.58 ± 0.08	0.58 ± 0.14	0.19 ± 0.05	**0.69 ± 0.04**
Insurances	0.48 ± 0.03	0.44 ± 0.03	0.42 ± 0.09	0.45 ± 0.06	0.48 ± 0.07	**0.58 ± 0.04**	0.59 ± 0.08	0.69 ± 0.05	0.64 ± 0.05	0.77 ± 0.04	0.35 ± 0.15	**0.86 ± 0.02**
Alarm	0.54 ± 0.05	0.57 ± 0.03	0.59 ± 0.06	0.56 ± 0.05	0.53 ± 0.06	**0.69 ± 0.04**	0.68 ± 0.04	0.82 ± 0.01	0.53 ± 0.04	0.79 ± 0.02	0.67 ± 0.02	**0.87 ± 0.02**
Child5	**0.56 ± 0.09**	0.22 ± 0.03	0.30 ± 0.04	0.39 ± 0.03	0.53 ± 0.05	0.50 ± 0.04	0.65 ± 0.03	0.68 ± 0.04	0.28 ± 0.03	0.69 ± 0.08	0.63 ± 0.02	**0.70 ± 0.02**
Insurances5	0.48 ± 0.02	0.44 ± 0.02	0.43 ± 0.03	0.48 ± 0.02	0.52 ± 0.03	**0.66 ± 0.02**	0.58 ± 0.02	0.65 ± 0.01	0.38 ± 0.02	0.75 ± 0.03	0.68 ± 0.01	**0.86 ± 0.01**
Alarm5	0.47 ± 0.02	0.48 ± 0.02	0.48 ± 0.03	0.51 ± 0.03	0.58 ± 0.04	**0.59 ± 0.01**	0.60 ± 0.01	0.72 ± 0.01	0.31 ± 0.02	0.72 ± 0.02	0.64 ± 0.02	**0.80 ± 0.02**
Child10	**0.55 ± 0.04**	0.23 ± 0.02	0.22 ± 0.02	0.40 ± 0.03	0.54 ± 0.03	0.50 ± 0.02	0.65 ± 0.01	0.67 ± 0.02	0.22 ± 0.01	**0.73 ± 0.07**	0.63 ± 0.05	0.72 ± 0.01
Insurances10	0.47 ± 0.01	0.43 ± 0.02	0.36 ± 0.01	0.47 ± 0.01	0.52 ± 0.03	**0.61 ± 0.01**	0.57 ± 0.03	0.67 ± 0.01	0.33 ± 0.01	0.75 ± 0.02	0.68 ± 0.01	**0.85 ± 0.01**
Alarm10	0.47 ± 0.02	0.48 ± 0.03	0.42 ± 0.02	0.52 ± 0.01	0.57 ± 0.02	**0.57 ± 0.01**	0.60 ± 0.01	0.68 ± 0.01	0.26 ± 0.01	0.73 ± 0.02	0.64 ± 0.01	**0.79 ± 0.01**
Hepar2	0.27 ± 0.10	0.14 ± 0.03	0.28 ± 0.05	0.30 ± 0.07	0.15 ± 0.08	**0.46 ± 0.05**	0.61 ± 0.04	0.40 ± 0.03	0.33 ± 0.04	0.64 ± 0.03	0.14 ± 0.06	**0.70 ± 0.03**
Win95pts	0.26 ± 0.03	0.18 ± 0.03	0.37 ± 0.03	0.43 ± 0.04	0.41 ± 0.04	**0.60 ± 0.03**	0.43 ± 0.02	0.54 ± 0.04	0.38 ± 0.03	0.65 ± 0.02	0.52 ± 0.02	**0.82 ± 0.01**
Andes	0.52 ± 0.02	0.52 ± 0.02	0.28 ± 0.01	0.66 ± 0.00	0.46 ± 0.02	**0.72 ± 0.00**	0.59 ± 0.02	0.74 ± 0.02	0.20 ± 0.01	0.79 ± 0.00	0.44 ± 0.02	**0.87 ± 0.00**
Link	0.24 ± 0.04	0.17 ± 0.03	0.23 ± 0.03	0.18 ± 0.03	0.18 ± 0.05	**0.30 ± 0.04**	0.50 ± 0.04	0.56 ± 0.07	0.38 ± 0.05	0.49 ± 0.03	0.23 ± 0.02	**0.56 ± 0.05**
w/t/l	10/0/3	13/0/0	13/0/0	13/0/0	11/0/2	——	13/0/0	13/0/0	13/0/0	12/0/1	13/0/0	——

Table 8
Recall (↑) results (Mean Recall ± Standard deviation) of $LCSL_{SF}$ and its rivals on static feature space with 500/5000 instances.

	500-instances						5000-instances					
Datasets	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$
Child	0.57 ± 0.09	0.31 ± 0.07	0.43 ± 0.22	0.53 ± 0.10	0.28 ± 0.17	**0.57 ± 0.05**	0.70 ± 0.04	0.63 ± 0.02	**0.73 ± 0.05**	0.60 ± 0.15	0.18 ± 0.04	0.71 ± 0.02
Insurances	0.39 ± 0.03	0.34 ± 0.03	0.28 ± 0.07	0.35 ± 0.05	0.34 ± 0.05	**0.47 ± 0.04**	0.51 ± 0.07	0.53 ± 0.04	0.62 ± 0.04	0.65 ± 0.03	0.26 ± 0.10	**0.73 ± 0.02**
Alarm	0.52 ± 0.04	0.52 ± 0.03	0.55 ± 0.04	0.52 ± 0.04	0.46 ± 0.05	**0.65 ± 0.03**	0.69 ± 0.04	0.79 ± 0.02	0.76 ± 0.04	0.81 ± 0.02	0.57 ± 0.02	**0.85 ± 0.02**
Child5	**0.56 ± 0.09**	0.23 ± 0.03	0.30 ± 0.04	0.39 ± 0.02	0.46 ± 0.04	0.48 ± 0.03	0.68 ± 0.03	0.68 ± 0.04	0.52 ± 0.03	0.72 ± 0.09	0.55 ± 0.02	**0.73 ± 0.02**
Insurances5	0.42 ± 0.02	0.35 ± 0.01	0.30 ± 0.02	0.41 ± 0.02	0.34 ± 0.03	**0.54 ± 0.01**	0.51 ± 0.02	0.56 ± 0.01	0.44 ± 0.02	0.70 ± 0.03	0.39 ± 0.01	**0.78 ± 0.01**
Alarm5	0.43 ± 0.03	0.39 ± 0.01	0.42 ± 0.03	0.44 ± 0.03	0.47 ± 0.04	**0.52 ± 0.01**	0.59 ± 0.01	0.62 ± 0.02	0.58 ± 0.02	0.70 ± 0.02	0.50 ± 0.01	**0.73 ± 0.01**
Child10	**0.56 ± 0.03**	0.23 ± 0.02	0.25 ± 0.02	0.40 ± 0.03	0.46 ± 0.03	0.49 ± 0.01	0.68 ± 0.02	0.66 ± 0.02	0.48 ± 0.01	**0.76 ± 0.07**	0.54 ± 0.04	0.74 ± 0.01
Insurances10	0.42 ± 0.01	0.34 ± 0.02	0.25 ± 0.01	0.41 ± 0.02	0.34 ± 0.02	**0.53 ± 0.01**	0.51 ± 0.02	0.57 ± 0.01	0.41 ± 0.01	0.70 ± 0.02	0.39 ± 0.00	**0.76 ± 0.01**
Alarm10	0.41 ± 0.01	0.36 ± 0.02	0.36 ± 0.02	0.43 ± 0.01	0.45 ± 0.01	**0.48 ± 0.01**	0.56 ± 0.01	0.56 ± 0.01	0.54 ± 0.01	0.69 ± 0.02	0.47 ± 0.01	**0.70 ± 0.01**
Hepar2	0.20 ± 0.08	0.08 ± 0.02	0.25 ± 0.04	0.22 ± 0.05	0.10 ± 0.05	**0.32 ± 0.05**	0.53 ± 0.04	0.31 ± 0.03	0.45 ± 0.03	0.56 ± 0.03	0.10 ± 0.04	**0.61 ± 0.02**
Win95pts	0.20 ± 0.02	0.11 ± 0.02	0.33 ± 0.02	0.32 ± 0.03	0.35 ± 0.03	**0.43 ± 0.02**	0.40 ± 0.02	0.42 ± 0.03	0.62 ± 0.04	0.59 ± 0.02	0.41 ± 0.02	**0.68 ± 0.01**
Andes	0.41 ± 0.02	0.39 ± 0.02	0.46 ± 0.02	0.57 ± 0.00	0.34 ± 0.02	**0.58 ± 0.00**	0.55 ± 0.03	0.63 ± 0.02	0.72 ± 0.01	0.78 ± 0.00	0.32 ± 0.01	**0.80 ± 0.00**
Link	0.29 ± 0.04	0.19 ± 0.03	0.18 ± 0.02	0.20 ± 0.03	0.20 ± 0.04	**0.31 ± 0.04**	0.36 ± 0.04	0.37 ± 0.04	0.29 ± 0.04	0.39 ± 0.03	0.22 ± 0.02	**0.39 ± 0.04**
w/t/l	11/0/2	13/0/0	13/0/0	13/0/0	13/0/0	——	13/0/0	13/0/0	12/0/1	12/0/1	13/0/0	——

Table 9
F1 (↑) results (Mean F1 ± Standard deviation) of $LCSL_{SF}$ and its rivals on static feature space with 500/5000 instances.

	500-instances						5000-instances					
Datasets	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$	CMB	PCD-by-PCD	MB-by-MB	ELCS	LCS-FS	$LCSL_{SF}$
Child	**0.57 ± 0.09**	0.29 ± 0.07	0.42 ± 0.23	0.53 ± 0.10	0.29 ± 0.18	0.56 ± 0.05	0.69 ± 0.04	0.63 ± 0.02	0.62 ± 0.07	0.59 ± 0.14	0.19 ± 0.04	**0.70 ± 0.03**
Insurances	0.42 ± 0.02	0.37 ± 0.03	0.33 ± 0.08	0.38 ± 0.05	0.39 ± 0.06	**0.50 ± 0.04**	0.54 ± 0.07	0.59 ± 0.04	0.61 ± 0.04	0.69 ± 0.03	0.29 ± 0.11	**0.77 ± 0.02**
Alarm	0.52 ± 0.04	0.53 ± 0.03	0.56 ± 0.05	0.53 ± 0.04	0.48 ± 0. 05	**0.66 ± 0.03**	0.68 ± 0.04	0.80 ± 0.02	0.60 ± 0.04	0.80 ± 0.02	0.61 ± 0.02	**0.86 ± 0.02**
Child5	**0.54 ± 0.09**	0.22 ± 0.03	0.28 ± 0.04	0.38 ± 0.02	0.48 ± 0.04	0.48 ± 0.03	0.66 ± 0.03	0.68 ± 0.04	0.35 ± 0.03	0.70 ± 0.09	0.58 ± 0.02	**0.71 ± 0.02**
Insurances5	0.43 ± 0.02	0.37 ± 0.02	0.33 ± 0.02	0.42 ± 0.02	0.39 ± 0.03	**0.57 ± 0.01**	0.53 ± 0.02	0.59 ± 0.01	0.40 ± 0.02	0.72 ± 0.03	0.48 ± 0.01	**0.81 ± 0.01**
Alarm5	0.44 ± 0.02	0.41 ± 0.01	0.43 ± 0.03	0.46 ± 0.03	0.50 ± 0.04	**0.53 ± 0.01**	0.59 ± 0.01	0.65 ± 0.02	0.38 ± 0.02	0.70 ± 0.01	0.55 ± 0.02	**0.75 ± 0.01**
Child10	**0.54 ± 0.03**	0.22 ± 0.02	0.21 ± 0.02	0.39 ± 0.03	0.49 ± 0.03	0.48 ± 0.02	0.66 ± 0.01	0.66 ± 0.02	0.29 ± 0.01	0.73 ± 0.07	0.57 ± 0.05	**0.73 ± 0.01**
Insurances10	0.42 ± 0.01	0.37 ± 0.02	0.28 ± 0.01	0.42 ± 0.01	0.39 ± 0.02	**0.55 ± 0.01**	0.53 ± 0.02	0.61 ± 0.01	0.35 ± 0.01	0.71 ± 0.02	0.47 ± 0.01	**0.79 ± 0.01**
Alarm10	0.42 ± 0.01	0.39 ± 0.02	0.36 ± 0.02	0.45 ± 0.01	0.49 ± 0.03	**0.49 ± 0.02**	0.57 ± 0.01	0.60 ± 0.01	0.33 ± 0.01	0.69 ± 0.02	0.52 ± 0.01	**0.73 ± 0.01**
Hepar2	0.21 ± 0.08	0.09 ± 0.02	0.24 ± 0.04	0.23 ± 0.05	0.11 ± 0.06	**0.34 ± 0.05**	0.55 ± 0.04	0.33 ± 0.02	0.35 ± 0.03	0.58 ± 0.03	0.11 ± 0.04	**0.63 ± 0.02**
Win95pts	0.22 ± 0.02	0.13 ± 0.02	0.32 ± 0.02	0.35 ± 0.03	0.37 ± 0.03	**0.48 ± 0.02**	0.40 ± 0.02	0.46 ± 0.03	0.43 ± 0.03	0.61 ± 0.03	0.45 ± 0.02	**0.72 ± 0.01**
Andes	0.44 ± 0.03	0.43 ± 0.02	0.32 ± 0.01	0.59 ± 0.00	0.37 ± 0.02	**0.62 ± 0.00**	0.56 ± 0.02	0.67 ± 0.02	0.29 ± 0.01	0.76 ± 0.00	0.36 ± 0.01	**0.82 ± 0.00**
Link	0.24 ± 0.03	0.17 ± 0.03	0.18 ± 0.02	0.18 ± 0.03	0.18 ± 0.04	**0.28 ± 0.04**	0.41 ± 0.04	0.42 ± 0.05	0.28 ± 0.04	0.42 ± 0.03	0.21 ± 0.02	**0.45 ± 0.04**
w/t/l	10/0/3	13/0/0	13/0/0	13/0/0	12/0/1	——	13/0/0	13/0/0	13/0/0	13/0/0	13/0/0	——



(a) SHD (↓) **(b)** Precision(↑)

(c) Recall (↑) **(d)** F1 (↑)

Fig. 5. Average performance of $LCSL_{SF}$ and its rivals on SHD, Precision, Recall and F1 with 500/5000 instances. The labels of the x-axis from 1 to 6 denote: 1:CMB; 2:PCD-by-PCD; 3:MB-by-MB; 4: ELCS; 5:LCS-FS; 6:$LCSL_{SF}$.

Table 10
Statistical results of $LCSL_{SF}$ and its rivals on SHD, Precision, Recall, and F1.

	Evaluation metric	χ_F^2	F_F	Critical value ($\alpha = 0.05$)
500-instances	SHD	34.99	14.00	1.914
	Precision	30.54	10.63	
	Recall	37.18	16.04	
	F1	35.80	14.71	
5000-instances	SHD	60.34	155.41	1.914
	Precision	50.26	40.93	
	Recall	46.48	30.12	
	F1	52.20	48.92	

1) For the above metrics, compared to CMB, PCD-by-PCD, MB-by-MB, ELCS, and LCS-FS, $LCSL_{SF}$ obtains better results on nine synthetic datasets with 500 instances. Notably: a) For the Alarm10 dataset, $LCSL_{SF}$ outperforms PCD-by-PCD on all evaluation metrics except for a difference of only 0.01 on SHD. b) On the Child, Child5, and Child10 datasets, $LCSL_{SF}$ is at a disadvantage compared to CMB, ELCS, and LCS-FS. This is because that in the context of streaming features, when the number of instances is small, the feature space including MB cannot be mined very accurately.

2) With the number of instances rising to 5000, $LCSL_{SF}$ achieves superiority or at least comparable performance to its rivals on all datasets. This is mainly due to two reasons, the first being that with increasing numbers of instances, a more complete MB can be mined. The second reason is to detect the V-structure of each feature in the aMB space.

3) Fig. 5 illustrates that the average performance of $LCSL_{SF}$ is superior to its rivals. In particular, $LCSL_{SF}$ obtains outstanding performances on the number of 500/5000 instances, where $LCSL_{SF}$ has the most significant improvement with the increasing of instances.

Overall, the experiments reveal that $LCSL_{SF}$ provides a competitive performance in other existing static LCSL algorithms.

Statistical Test. Table 10 summarizes the χ_F^2 and F_F results of SHD, Precision, Recall, and F1 metrics under 500/5000 instances, respectively. As shown in Table 10, we conduct the Friedman test at the 5% significance level under the null hypothesis, which states that there is significant difference between the performance of $LCSL_{SF}$ and that of its rivals. Then, we proceed with the Nemenyi test

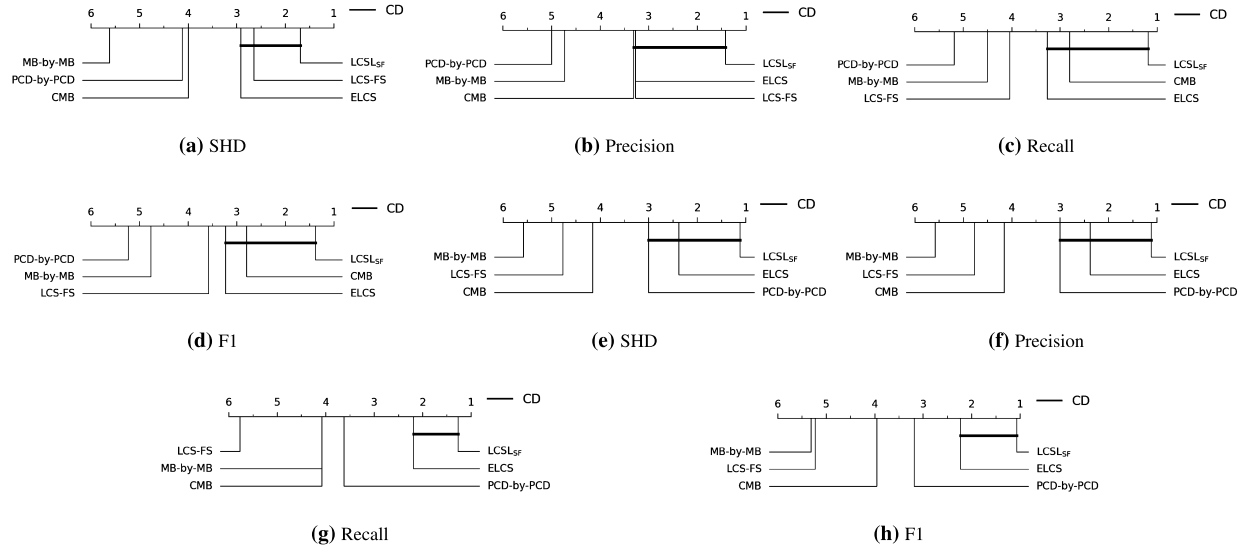


Fig. 6. CD diagram of Nemenyi test on different evaluation metrics about LCSL$_{SF}$ and its rivals in static feature space.

as a post hoc test. For the Nemenyi test, the critical difference is 2.09 (where $q_\alpha = 2.850$ at a significance level $alpha = 0.05$, $N = 13$, $k = 6$). According to the average rank and critical difference, CD diagrams are constructed, as shown in Fig. 6.

From Figs. 6(a)-(d), we observe that: 1) For all evaluation metrics, LCSL$_{SF}$ outperforms PCD-by-PCD and MB-by-MB; 2) The results for F1 and Recall show that LCSL$_{SF}$ exhibits statistically superior performance compared to LCS-FS; 3) Compared to CMB, LCSL$_{SF}$ exhibits better performance on the SHD metric and outperforms remaining rivals in statistical rankings for the other three evaluation metrics; 4) On the four evaluation metrics, LCSL$_{SF}$ and ELCS are connected on the CD diagram, indicating that there is no significant difference between LCSL$_{SF}$ and ELCS. However, LCSL$_{SF}$ ranks closer to one, and LCSL$_{SF}$ is slightly better than ELCS.

Figs. 6(e)-(f) show that: 1) LCSL$_{SF}$ is more significantly outstanding than CMB, MB-by-MB, and LCS-FS algorithms with respect to the SHD, F1, Precision, and Recall results; 2) For F1 and Recall, LCSL$_{SF}$ is likewise statistically superior to PCD-by-PCD; 3) On the four evaluation metrics, LCSL$_{SF}$ and ELCS are connected. Compared to ELCS on four evaluation metrics, although the CD diagrams show no difference between the two algorithms, LCSL$_{SF}$ ranks closer to one in statistics than ELCS, and LCSL$_{SF}$ constructs local causal structures in the dynamic feature space, such that LCSL$_{SF}$ has an advantage over ELCS. Overall, LCSL$_{SF}$ obtains excellent performance than its competitors.

Stability analysis. In this study, to verify the stability of algorithms, we draw spider web diagrams to illustrate the stability index in terms of each evaluation metric. All metrics are normalized into [0.1, 0.5] as a general standard.

From Fig. 7, we observe that: 1) For Precision, Recall, and F1 on synthetic datasets with 500 instances, the shape of LCSL$_{SF}$ is very close to the whole area, which means that LCSL$_{SF}$ obtains a more stable solution. For SHD, LCSL$_{SF}$ remains stable on at least ten datasets and obtains a slightly more stable solution than other algorithms; 2) For Precision, Recall, and F1 on synthetic datasets with 5000 instances, LCSL$_{SF}$ shows better stability, and achieves the more stable solution with the increasing instances.

In summary, in comparison to state-of-the-art static LCSL algorithms, LCSL$_{SF}$ maintains excellent performance on static feature space duo such that LCSL$_{SF}$ can accurately mine causal features and distinguish direct causes/effects.

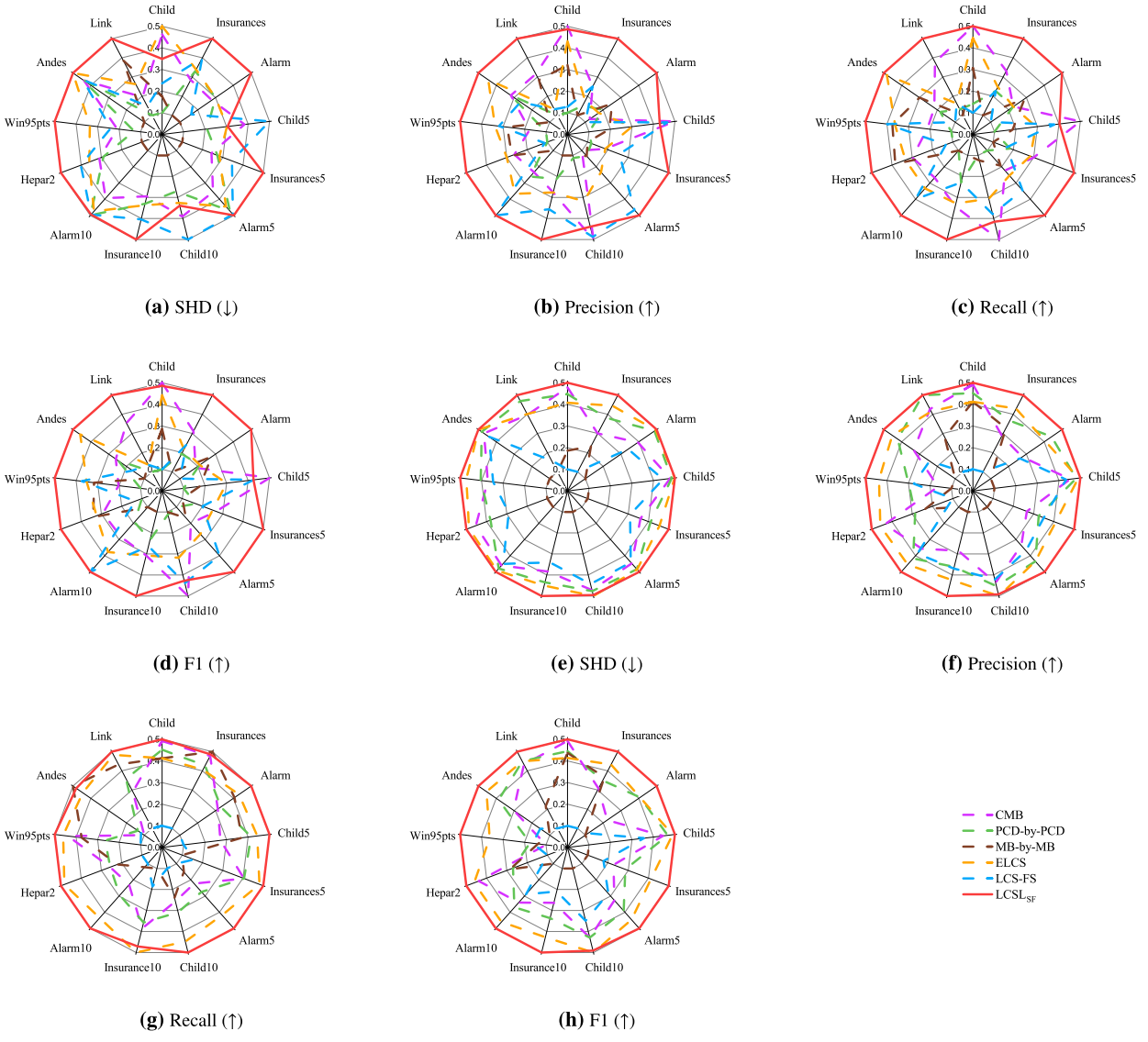
5.4. Comparison with LCSL algorithms for dynamic feature space (RQ3)

To further analyze the performance of LCSL$_{SF}$ on constructing local causal structure in the dynamic feature space, we propose three variants of LCSL$_{SF}$, named LCSL$_{OSFS}$, LCSL$_{SAOLA}$, and LCSL$_{OCFSSF}$ with different alternatives of OL$_{aMB}$ by OSFS, SAOLA, and OCFSSF to mine causal features, respectively. We run these algorithms on the dynamic feature space with streaming features simulated by synthetic datasets.

Performance evaluation. Tables 11-14 and Fig. 8 present the evaluation results of LCSL$_{SF}$ and other three competitors on the metrics of SHD, Precision, Recall, and F1. The observations are as follows.

1) In terms of the above metrics, LCSL$_{SF}$ clearly outperforms LCSL$_{OSFS}$ and LCSL$_{OCFSSF}$ in at least ten datasets with 500 instances. Notably LCSL$_{SF}$ outperforms LCSL$_{SAOLA}$ on at least eight datasets. This is because under the streaming feature, due to the small number of instances, the aMB mined by LCSL$_{SF}$ on the Child, Child5, Child10, and Link datasets only contain part of MB.

2) Especially using 5,000 instances on the SHD, F1, Precision, and Recall, LCSL$_{SF}$ is significantly better than LCSL$_{OSFS}$ and LCSL$_{SAOLA}$ on all datasets, and achieves superior or at least comparable performance compared to LCSL$_{OCFSSF}$ on all datasets. This is because LCSL$_{SF}$ can learn a more accurate MB as the number of instances increases, whereas OSFS learns only PC of the target variable and SAOLA learns feature sets with redundant features, while LCSL$_{OCFSSF}$ learns only a partial MB of the target variable. Compared with LCSL$_{SF}$, the three competitors present lower performance on the above metrics due to mined MB with incomplete features.



(a) SHD (↓) **(b)** Precision (↑) **(c)** Recall (↑)

(d) F1 (↑) **(e)** SHD (↓) **(f)** Precision (↑)

(g) Recall (↑) **(h)** F1 (↑)

Fig. 7. Spider web diagrams for stability analysis.

3) From Fig. 8, we find that the average performance of LCSL_{SF} performs better than its rivals. In comparison, LCSL_{SF} achieves outstanding performances on the number of 500/5000 instances.

Statistical Test. To further analyze the performance of the LCSL_{SF}, $\text{LCSL}_{\text{OSFS}}$, $\text{LCSL}_{\text{SAOLA}}$, and $\text{LCSL}_{\text{OCFSSF}}$ algorithms, Table 15 lists χ_F^2 and F_F in terms of SHD, Precision, Recall, F1 under 500/5000 instances, respectively. Using the Friedman test at the 5% significance level to evaluate the performance of the LCSL_{SF} and its rivals on four metrics, the null hypothesis is rejected. From the Nemenyi test, the CD = 1.30 (where $q_\alpha = 2.569$ at a significance level $alpha = 0.05$, $N = 13$, $k = 4$), and the comparison between LCSL_{SF} and other algorithms is displayed in Fig. 9.

Figs. 9(a)-(d) show that: 1) LCSL_{SF} performs better than $\text{LCSL}_{\text{OSFS}}$ and $\text{LCSL}_{\text{OCFSSF}}$ for all evaluation metrics, i.e., SHD, precision, Recall, and F1; 2) In comparison to $\text{LCSL}_{\text{SAOLA}}$, LCSL_{SF} and $\text{LCSL}_{\text{SAOLA}}$ are connected on the CD diagrams for the four evaluation metrics, indicating that there is no significant difference between LCSL_{SF} and $\text{LCSL}_{\text{SAOLA}}$. However, LCSL_{SF} ranks closer to one, LCSL_{SF} performs slightly better than $\text{LCSL}_{\text{SAOLA}}$.

Based on Figs. 9(e)-(h), we can conclude that: 1) LCSL_{SF} yield better performance than $\text{LCSL}_{\text{OSFS}}$ and $\text{LCSL}_{\text{SAOLA}}$ algorithms in terms of SHD, precision, Recall, and F1; 2) LCSL_{SF} and $\text{LCSL}_{\text{OCFSSF}}$ are comparable on the four evaluation metrics, although the CD diagram shows no difference between the two algorithms. LCSL_{SF} ranks statistically closer to one than $\text{LCSL}_{\text{OCFSSF}}$, such that LCSL_{SF} has an advantage over $\text{LCSL}_{\text{OCFSSF}}$. Overall, LCSL_{SF} obtains excellent performance when compared with the other methods.

Table 11
SHD (↓) results (Mean SHD ± Standard deviation) of $LCSL_{SF}$ and its rivals on dynamic feature space with 500/5000 instances.

	500-instances				5000-instances			
Datasets	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$
Child	2.09 ± 0.11	**1.34 ± 0.13**	1.55 ± 0.19	1.68 ± 0.16	1.13 ± 0.04	1.41 ± 0.04	0.97 ± 0.12	**0.78 ± 0.09**
Insurances	3.84 ± 0.16	2.82 ± 0.12	3.05 ± 0.16	**2.79 ± 0.15**	2.53 ± 0.06	2.70 ± 0.08	2.31 ± 0.08	**1.40 ± 0.08**
Alarm	1.89 ± 0.07	1.53 ± 0.09	2.67 ± 0.13	**1.16 ± 0.10**	1.30 ± 0.07	1.15 ± 0.07	1.57 ± 0.11	**0.41 ± 0.09**
Child5	1.77 ± 0.05	**1.54 ± 0.04**	2.30 ± 0.08	1.98 ± 0.13	1.22 ± 0.03	1.43 ± 0.04	1.11 ± 0.05	**0.77 ± 0.06**
Insurances5	3.85 ± 0.04	2.79 ± 0.05	3.39 ± 0.06	**2.58 ± 0.08**	2.54 ± 0.04	2.40 ± 0.03	1.97 ± 0.08	**1.28 ± 0.06**
Alarm5	2.15 ± 0.05	1.93 ± 0.04	2.90 ± 0.03	**1.90 ± 0.06**	1.63 ± 0.03	1.62 ± 0.02	1.70 ± 0.04	**0.97 ± 0.05**
Child10	1.84 ± 0.03	**1.66 ± 0.04**	2.34 ± 0.03	2.04 ± 0.08	2.49 ± 0.01	1.46 ± 0.03	1.14 ± 0.04	**0.76 ± 0.04**
Insurances10	3.76 ± 0.04	2.74 ± 0.05	3.33 ± 0.04	**2.70 ± 0.06**	2.49 ± 0,01	2.47 ± 0.03	1.92 ± 0.04	**1.31 ± 0.04**
Alarm10	2.42 ± 0.03	2.24 ± 0.05	3.18 ± 0.04	**2.19 ± 0.05**	1.79 ± 0.03	1.76 ± 0.03	1.93 ± 0.04	**1.13 ± 0.04**
Hepar2	3.23 ± 0.11	**3.09 ± 0.14**	3.19 ± 0.11	3.11 ± 0.15	2.16 ± 0.08	2.09 ± 0.04	1.96 ± 0.06	**1.94 ± 0.07**
Win95pts	2.89 ± 0.04	2.58 ± 0.07	2.18 ± 0.07	**2.11 ± 0.07**	2.20 ± 0.07	2.13 ± 0.06	1.36 ± 0.06	**1.28 ± 0.06**
Andes	2.14 ± 0.00	2.23 ± 0.00	1.79 ± 0.00	**1.78 ± 0.00**	1.27 ± 0.00	1.52 ± 0.00	**0.94 ± 0.00**	**0.94 ± 0.00**
Link	**3.43 ± 0.31**	3.48 ± 0.32	3.48 ± 0.28	4.00 ± 0.39	2.34 ± 0.22	3.29 ± 0.45	2.05 ± 0.33	**2.05 ± 0.18**
w/t/l	10/0/3	8/0/5	11/0/2	——	13/0/0	13/0/0	12/1/0	——

Table 12
Precision (↑) results (Mean Precision ± Standard deviation) of $LCSL_{SF}$ and its rivals on dynamic feature space with 500/5000 instances.

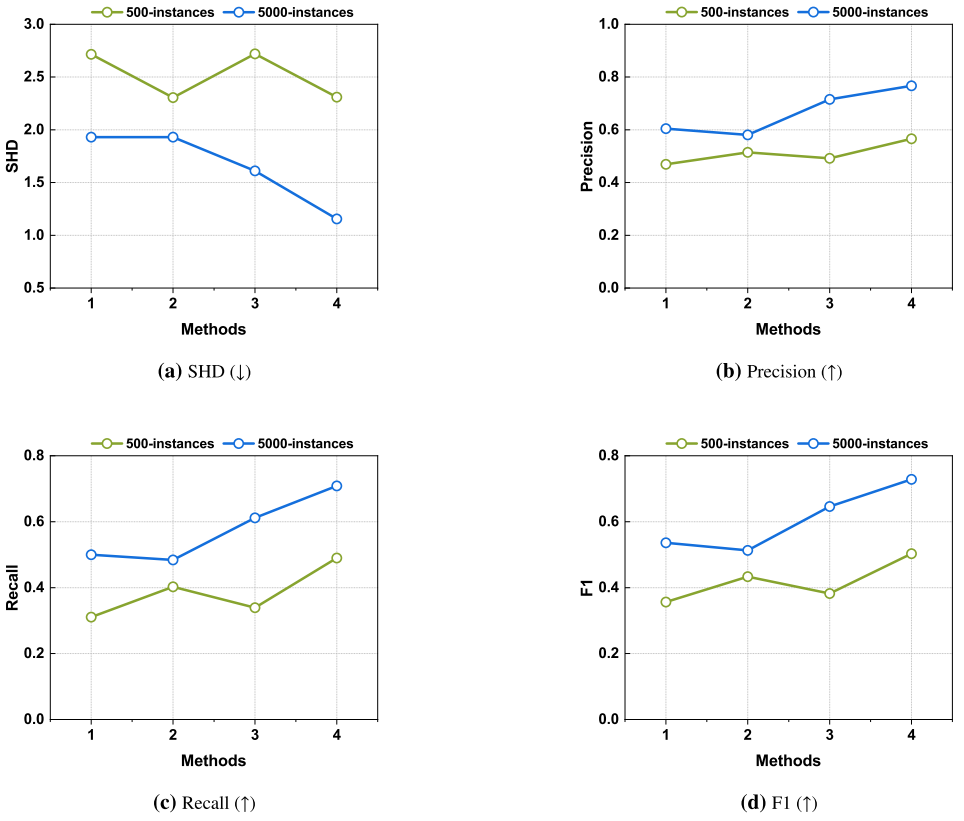
	500-instances				5000-instances			
Datasets	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$
Child	0.52 ± 0.03	**0.67 ± 0.04**	0.63 ± 0.07	0.58 ± 0.06	0.63 ± 0.01	0.58 ± 0.01	0.66 ± 0.05	**0.69 ± 0.04**
Insurances	0.36 ± 0.04	0.57 ± 0.03	0.58 ± 0.06	**0.58 ± 0.04**	0.66 ± 0.01	0.59 ± 0.03	0.73 ± 0.02	**0.86 ± 0.02**
Alarm	0.40 ± 0.02	0.48 ± 0.03	0.34 ± 0.04	**0.69 ± 0.04**	0.53 ± 0.02	0.58 ± 0.02	0.67 ± 0.03	**0.87 ± 0.02**
Child5	**0.71 ± 0.01**	0.59 ± 0.02	0.47 ± 0.03	0.50 ± 0.04	0.60 ± 0.01	0.60 ± 0.01	0.63 ± 0.01	**0.70 ± 0.02**
Insurances5	0.49 ± 0.01	0.64 ± 0.02	0.64 ± 0.02	**0.66 ± 0.02**	0.71 ± 0.01	0.71 ± 0.01	0.84 ± 0.01	**0.86 ± 0.01**
Alarm5	0.51 ± 0.01	0.51 ± 0.01	0.36 ± 0.02	**0.59 ± 0.01**	0.57 ± 0.01	0.56 ± 0.01	0.68 ± 0.01	**0.80 ± 0.02**
Child10	**0.70 ± 0.02**	0.56 ± 0.01	0.47 ± 0.01	0.50 ± 0.02	0.70 ± 0.00	0.60 ± 0.01	0.65 ± 0.01	**0.72 ± 0.01**
Insurances10	0.48 ± 0.02	**0.61 ± 0.01**	0.61 ± 0.02	**0.61 ± 0.01**	0.70 ± 0.01	0.68 ± 0.01	0.83 ± 0.01	**0.85 ± 0.01**
Alarm10	0.51 ± 0.01	0.50 ± 0.01	0.33 ± 0.01	**0.57 ± 0.01**	0.58 ± 0.01	0.58 ± 0.01	0.67 ± 0.01	**0.79 ± 0.01**
Hepar2	0.44 ± 0.05	**0.46 ± 0.05**	0.43 ± 0.05	**0.46 ± 0.05**	0.69 ± 0.02	0.69 ± 0.02	**0.72 ± 0.02**	0.70 ± 0.03
Win95pts	0.26 ± 0.02	0.30 ± 0.02	0.58 ± 0.04	**0.60 ± 0.03**	0.41 ± 0.02	0.41 ± 0.01	0.79 ± 0.01	**0.82 ± 0.01**
Andes	0.49 ± 0.00	0.49 ± 0.00	**0.72 ± 0.00**	**0.72 ± 0.00**	0.60 ± 0.00	0.57 ± 0.00	**0.87 ± 0.00**	**0.87 ± 0.00**
Link	0.23 ± 0.04	0.30 ± 0.05	0.23 ± 0.05	**0.30 ± 0.04**	0.48 ± 0.07	0.37 ± 0.03	0.56 ± 0.06	**0.56 ± 0.05**
w/t/l	11/0/2	8/2/3	12/1/0	——	13/0/0	13/0/0	11/1/1	——

Table 13
Recall (↑) results (Mean Recall ± Standard deviation) of $LCSL_{SF}$ and its rivals on dynamic feature space with 500/5000 instances.

	500-instances				5000-instances			
Datasets	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$
Child	0.41 ± 0.03	0.54 ± 0.04	0.51 ± 0.05	**0.57 ± 0.05**	0.59 ± 0.01	0.49 ± 0.02	0.65 ± 0.03	**0.71 ± 0.02**
Insurances	0.22 ± 0.03	0.39 ± 0.02	0.35 ± 0.03	**0.47 ± 0.04**	0.50 ± 0.01	0.43 ± 0.02	0.54 ± 0.01	**0.73 ± 0.02**
Alarm	0.32 ± 0.03	0.42 ± 0.03	0.28 ± 0.04	**0.65 ± 0.03**	0.50 ± 0.02	0.52 ± 0.01	0.62 ± 0.02	**0.85 ± 0.02**
Child5	0.46 ± 0.01	**0.49 ± 0.01**	0.38 ± 0.01	0.48 ± 0.03	0.57 ± 0.01	0.49 ± 0.01	0.63 ± 0.01	**0.73 ± 0.02**
Insurances5	0.28 ± 0.01	0.48 ± 0.01	0.33 ± 0.01	**0.54 ± 0.01**	0.54 ± 0.01	0.54 ± 0.01	0.65 ± 0.01	**0.78 ± 0.01**
Alarm5	0.34 ± 0.01	0.42 ± 0.01	0.26 ± 0.01	**0.52 ± 0.01**	0.50 ± 0.01	0.49 ± 0.01	0.59 ± 0.01	**0.73 ± 0.01**
Child10	0.45 ± 0.01	**0.49 ± 0.01**	0.37 ± 0.01	**0.49 ± 0.01**	0.54 ± 0.00	0.52 ± 0.01	0.63 ± 0.01	**0.74 ± 0.01**
Insurances10	0.27 ± 0.01	0.48 ± 0.01	0.33 ± 0.01	**0.53 ± 0.01**	0.54 ± 0.01	0.53 ± 0.00	0.65 ± 0.01	**0.76 ± 0.01**
Alarm10	0.32 ± 0.01	0.40 ± 0.01	0.21 ± 0.02	**0.48 ± 0.01**	0.49 ± 0.01	0.50 ± 0.00	0.54 ± 0.01	**0.70 ± 0.01**
Hepar2	0.29 ± 0.03	**0.33 ± 0.04**	0.29 ± 0.03	0.32 ± 0.05	0.56 ± 0.02	0.56 ± 0.01	0.60 ± 0.02	**0.61 ± 0.02**
Win95pts	0.17 ± 0.01	0.20 ± 0.02	0.40 ± 0.02	**0.43 ± 0.02**	0.32 ± 0.01	0.31 ± 0.01	0.67 ± 0.01	**0.68 ± 0.02**
Andes	0.35 ± 0.00	0.37 ± 0.00	0.56 ± 0.00	**0.58 ± 0.00**	0.51 ± 0.00	0.48 ± 0.00	**0.80 ± 0.00**	**0.80 ± 0.00**
Link	0.16 ± 0.03	0.22 ± 0.04	0.14 ± 0.03	**0.31 ± 0.04**	0.34 ± 0.04	0.28 ± 0.03	0.38 ± 0.04	**0.39 ± 0.04**
w/t/l	13/0/0	10/1/2	13/0/0	——	13/0/0	13/0/0	12/1/0	——

Table 14
F1 (↑) results (Mean F1 ± Standard deviation) of $LCSL_{SF}$ and its rivals on dynamic feature space with 500/5000 instances.

	500-instances				5000-instances			
Datasets	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$	$LCSL_{OSFS}$	$LCSL_{SAOLA}$	$LCSL_{OCFSSF}$	$LCSL_{SF}$
Child	0.45 ± 0.03	**0.58 ± 0.04**	0.55 ± 0.06	0.56 ± 0.05	0.61 ± 0.01	0.53 ± 0.01	0.65 ± 0.04	**0.70 ± 0.03**
Insurances	0.26 ± 0.04	0.45 ± 0.02	0.42 ± 0.04	**0.50 ± 0.04**	0.56 ± 0.01	0.49 ± 0.02	0.61 ± 0.02	**0.77 ± 0.02**
Alarm	0.35 ± 0.03	0.44 ± 0.03	0.30 ± 0.04	**0.66 ± 0.03**	0.51 ± 0.02	0.54 ± 0.01	0.64 ± 0.03	**0.86 ± 0.02**
Child5	**0.53 ± 0.01**	0.51 ± 0.02	0.40 ± 0.01	0.48 ± 0.03	0.58 ± 0.01	0.52 ± 0.01	0.63 ± 0.01	**0.71 ± 0.02**
Insurances5	0.34 ± 0.01	0.53 ± 0.01	0.42 ± 0.01	**0.57 ± 0.01**	0.60 ± 0.01	0.60 ± 0.01	0.72 ± 0.01	**0.81 ± 0.01**
Alarm5	0.39 ± 0.01	0.44 ± 0.01	0.28 ± 0.01	**0.53 ± 0.01**	0.52 ± 0.01	0.51 ± 0.01	0.62 ± 0.01	**0.75 ± 0.01**
Child10	**0.52 ± 0.01**	0.50 ± 0.01	0.40 ± 0.01	0.48 ± 0.02	0.60 ± 0.00	0.54 ± 0.01	0.63 ± 0.01	**0.73 ± 0.01**
Insurances10	0.33 ± 0.01	0.53 ± 0.01	0.41 ± 0.01	**0.55 ± 0.01**	0.60 ± 0.01	0.58 ± 0.01	0.71 ± 0.01	**0.79 ± 0.01**
Alarm10	0.37 ± 0.01	0.43 ± 0.01	0.24 ± 0.01	**0.49 ± 0.02**	0.52 ± 0.01	0.53 ± 0.01	0.58 ± 0.01	**0.73 ± 0.01**
Hepar2	0.32 ± 0.04	0.35 ± 0.04	0.32 ± 0.03	**0.34 ± 0.05**	0.59 ± 0.02	0.59 ± 0.01	0.63 ± 0.03	**0.63 ± 0.02**
Win95pts	0.20 ± 0.01	0.23 ± 0.02	0.45 ± 0.03	**0.48 ± 0.02**	0.35 ± 0.01	0.34 ± 0.01	0.71 ± 0.01	**0.72 ± 0.01**
Andes	0.39 ± 0.00	0.41 ± 0.00	0.61 ± 0.00	**0.62 ± 0.00**	0.54 ± 0.00	0.51 ± 0.00	**0.82 ± 0.00**	**0.82 ± 0.00**
Link	0.18 ± 0.03	0.23 ± 0.04	0.17 ± 0.03	**0.28 ± 0.04**	0.39 ± 0.05	0.31 ± 0.03	0.44 ± 0.04	**0.45 ± 0.04**
w/t/l	11/0/2	10/0/3	13/0/0	——	13/0/0	13/0/0	12/1/0	——



(a) SHD (↓) **(b)** Precision (↑)

(c) Recall (↑) **(d)** F1 (↑)

Fig. 8. Average performance of $LCSL_{SF}$ and its rivals on SHD, F1, Precision, and Recall with 500/5000 instances. The labels of the x-axis from 1 to 4 denote the method: 1:$LCSL_{OSFS}$; 2:$LCSL_{SAOLA}$; 3:$LCSL_{OCFSSF}$; 4:$LCSL_{SF}$.

Table 15
Statistical results on $LCSL_{SF}$ and its rivals on SHD, Precision, Recall, and F1.

	Evaluation metric	χ_F^2	F_F	Critical value ($\alpha = 0.05$)
500-instances	SHD	12.72	5.81	2.0327
	Precision	9.44	3.83	
	Recall	28.92	134.41	
	F1	19.80	12.38	
5000-instances	SHD	25.34	22.26	2.0327
	Precision	32.63	61.48	
	Recall	434.55	93.08	
	F1	33.72	76.56	

Stability analysis. We perform the same stability analysis for $LCSL_{OSFS}$, $LCSL_{SAOLA}$, $LCSL_{OCFSSF}$, and $LCSL_{SF}$, normalizing the metrics to [0.1,0.5] as a general standard.

Fig. 10 shows that: 1) For Recall on synthetic datasets of 500 instances, $LCSL_{SF}$ obtains a more stable solution on eleven datasets. In terms of Precision and F1, $LCSL_{SF}$ remains stable on at least nine datasets. For SHD, $LCSL_{SF}$ achieves quite a stable solution with $LCSL_{SAOLA}$; however it performs also better than other algorithms; 2) On synthetic datasets of 5000 instances, SHD, F1, and Recall of $LCSL_{SF}$ show optimal stability, while for Precision, $LCSL_{SF}$ has a stable performance on the twelve datasets, and the enclosed area is the largest. Thus, in comparison to its three variants, $LCSL_{SF}$ performs more stably on the dynamic feature space.

Based on the above analysis, $LCSL_{SF}$ is more accurate in constructing local causal structures on the dynamic feature space than $LCSL_{OSFS}$, $LCSL_{SAOLA}$, and $LCSL_{OCFSSF}$.

5.5. Performance analysis with features arriving and sequence changes

To further confirm the effect of the new arrival features and sequence changes on learning LCS, we selected three benchmark datasets, e.g. Child, Insurance, and Alarm, including 20, 27, and 37 features, respectively. We set the continuous arriving features through original or random sequences. Figs. 11-13 show the performance on F1 with features continuous arriving and sequence

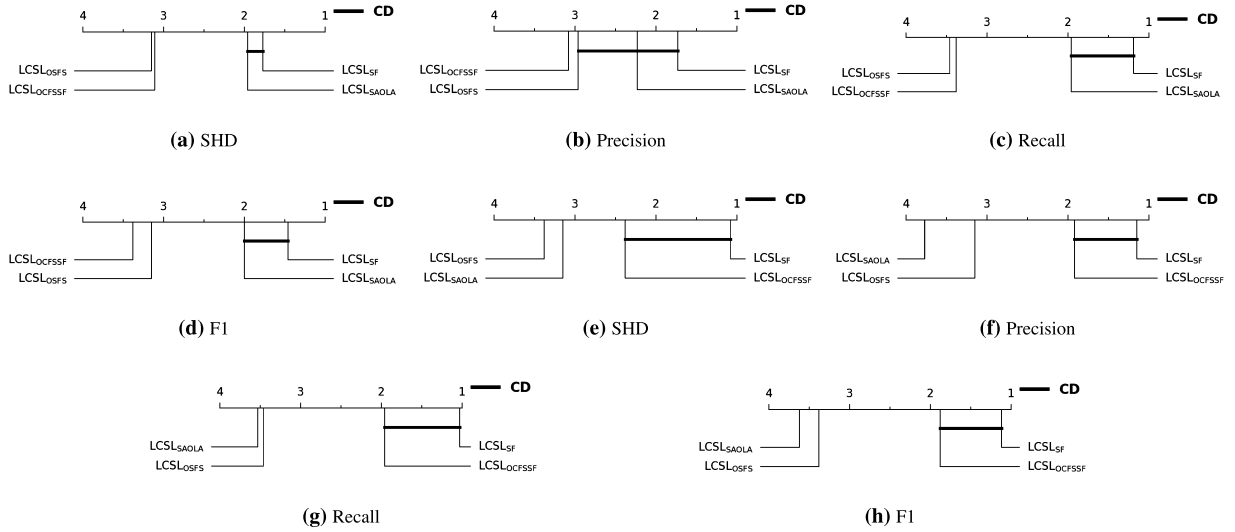


Fig. 9. CD diagram of Nemenyi test on different evaluation metrics about LCSL_{SF} and its rivals in dynamic feature space.

changes on the above three datasets. Due to page limitations, we only present the first 16 features of each dataset as the target variable.

From Figs. 11-13, we can observe that, 1) As the features continue to arrive, the indicator F1 continues to increase. This reflects that the causal structure learned by LCSL_{SF} becomes more accurate as the features increase; 2) The orders in which features appear are different in original and random sequences. This leads to differences in F1 during the continuous arrival of features; 3) When the arrived features are the same, F1 tends to be consistent. For example, when the last feature of each dataset arrives, the features of both sequences have already been received, and the features in the sequences are identical, so the final F1 is mostly equal; 4)For Figs. 11(j), 12(b, k, o), and 13(m), there is a difference in the final F1. This is because the cyclicity between features within the local structure leads to the influence of feature orders on the results; 5) From Fig. 11(a), the LCS of feature 1 is not accurately distinguished, and LCSL_{SF} incorrectly judges the child node of feature 1 as the parent node when identifying the direction since the causal features mined in OL_{aMB} contain only feature 2, and the true LCS has only one child node, i.e., feature 2, the LCSL_{SF} algorithm has the potential to identify the V-structure through multi-conditional independence with false positives.

In general, on the three benchmark datasets, regardless of the original or random arriving sequence of features, F1 gradually stabilizes. This means that under the independent identically distribution of data assumption, the causal structure of streaming features is essentially the same for both the original and random sequences.

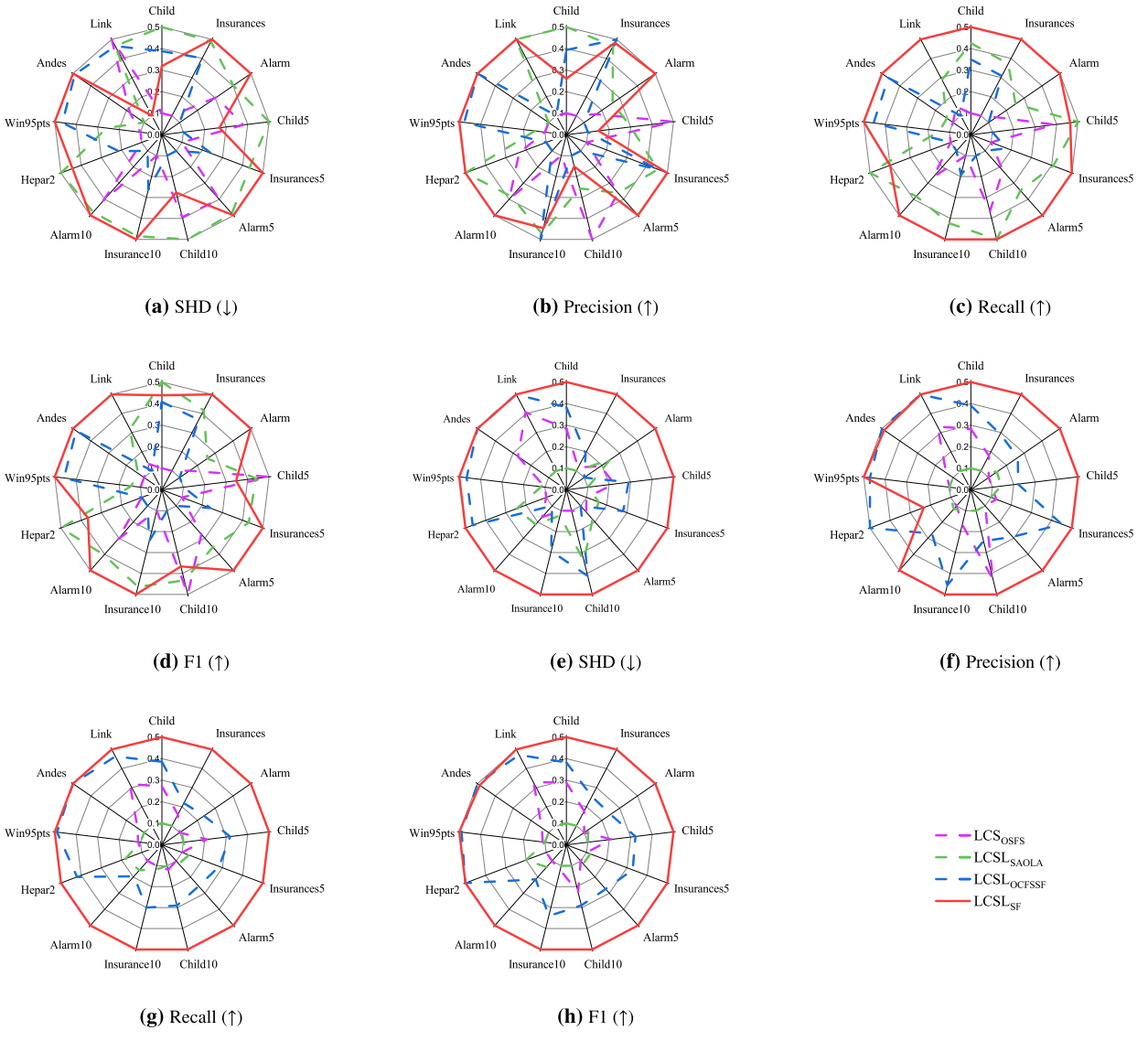
5.6. Discussion on the impact of hidden variables / not received features

For streaming feature scenarios, new features continuously arrive over time. We refer to the features that have not been received and remain unobservable in the current situation as hidden features, a.k.a., hidden variables. Here, we preliminarily discuss how hidden variables affect our LCSL_{SF} to generate LCS from two aspects of making for ascent performance and descent performance.

About ascent performance. From the original sequences (red lines) of Figs. 11 - 13, we can observe that the F1 curves maintain overall steady upward trends. It means that the true LCS is gradually revealed with the arrival of hidden variables. As shown in Fig. 14 (a), the f_{19} initially is the direct cause of the target feature f_7. When the f_{22} arrives, the f_{19} becomes the indirect cause (grandfather) of f_7. According to the rule of conditional independence test, we move f_{19} and retain f_{22}, as shown in Fig. 14 (b). Similarly, f_{25} replaces f_8 as a direct effect (child) of f_7. Therefore, the F1 curves rise because the direct cause (f_{22}) or effect (f_{25}) of f_7 is further discovered.

About descent performance. From the random sequences (orange lines) of Figs. 11 - 13, we can see that the F1 curves emerge more frequently fluctuate than the original sequence. This is because some arriving features deteriorate performance due to the possible hidden impacts. One of the important reasons is that the cause-and-effect sequences may be reversed in the random sequences. But, cause and effect mean that one event causes another event to occur, i.e., causes occur before effects in time. If the effect features arrive earlier than the cause ones, the curves of F1 maybe decline. This is because when the relationship between the arrived effect features and the features in aMB cannot be established, the causal evidence of subsequent arriving features will be cut off. The detailed analyses are as below.

- *Descent performance due to unfiltered redundant features.* Hidden variables may result in redundant features not being filtered from aMB. From Figs. 15, f_{15} and f_{14} are the redundant features of f_2. The f_{15} is the cause of effect f_{14}, i.e., $f_{15} \rightarrow f_{14}$. When the hidden variable f_{14} and f_{15} arrive sequentially, as shown in Fig. 15 (a), the f_{14} is discarded using d-separation among f_2, f_5 and



(a) SHD (↓) **(b)** Precision (↑) **(c)** Recall (↑)

(d) F1 (↑) **(e)** SHD (↓) **(f)** Precision (↑)

(g) Recall (↑) **(h)** F1 (↑)

Fig. 10. Spider web diagrams for stability analysis.

f_{14}. Then, when f_{15} is initially merged into the aMB, f_{15} cannot be filtered out by d-separation due to the missing of removed f_{14}, as shown in Fig. 15 (b). Therefore, the indicator F1 decrease, as shown in Figs. 11 - 13 (orange lines). In contrast, if f_{15} arrives before f_{14}, f_{15} and f_{14} will be filtered out using d-separation.

- *Descent performance due to removed causal features.* Hidden variables may result in causal features being filtered out. From Fig. 16, f_{17} and f_{10} are the direct cause and effect of the target feature f_{14}, respectively. f_3 is the direct cause of f_{10}, i.e., $f_3 \rightarrow f_{10}$. When f_{10} arrives before f_{17}, as shown in Fig. 16 (b)-(d), f_3 will be removed due to the d-separation and backdoor criterion among f_{10}, f_3 and f_{14}. The removal of f_3 causes the decrease of F1 because it is a causal feature. Otherwise, if f_{17} arrives before f_{10}, as shown in Fig. 16 (b′)-(d′), the causal feature f_3 will not be discarded. Although the arrival sequence of features may lead to the decrease of F1, removing f_3 is difficult when it maintains the cause-and-effect with other features of aMB.

To sum up, although some of the hidden variables / not received features can deteriorate performance, the deterioration should occur under relatively harsh situations. For example, in Fig. 15(b), if f_{15} has causal edges with the features in aMB except for f_{14}, the redundant feature f_{15} may be removed from aMB, without resulting in F1 decrease. In real scenarios, LCSL_{SF} can usually remain stable and rising performance.

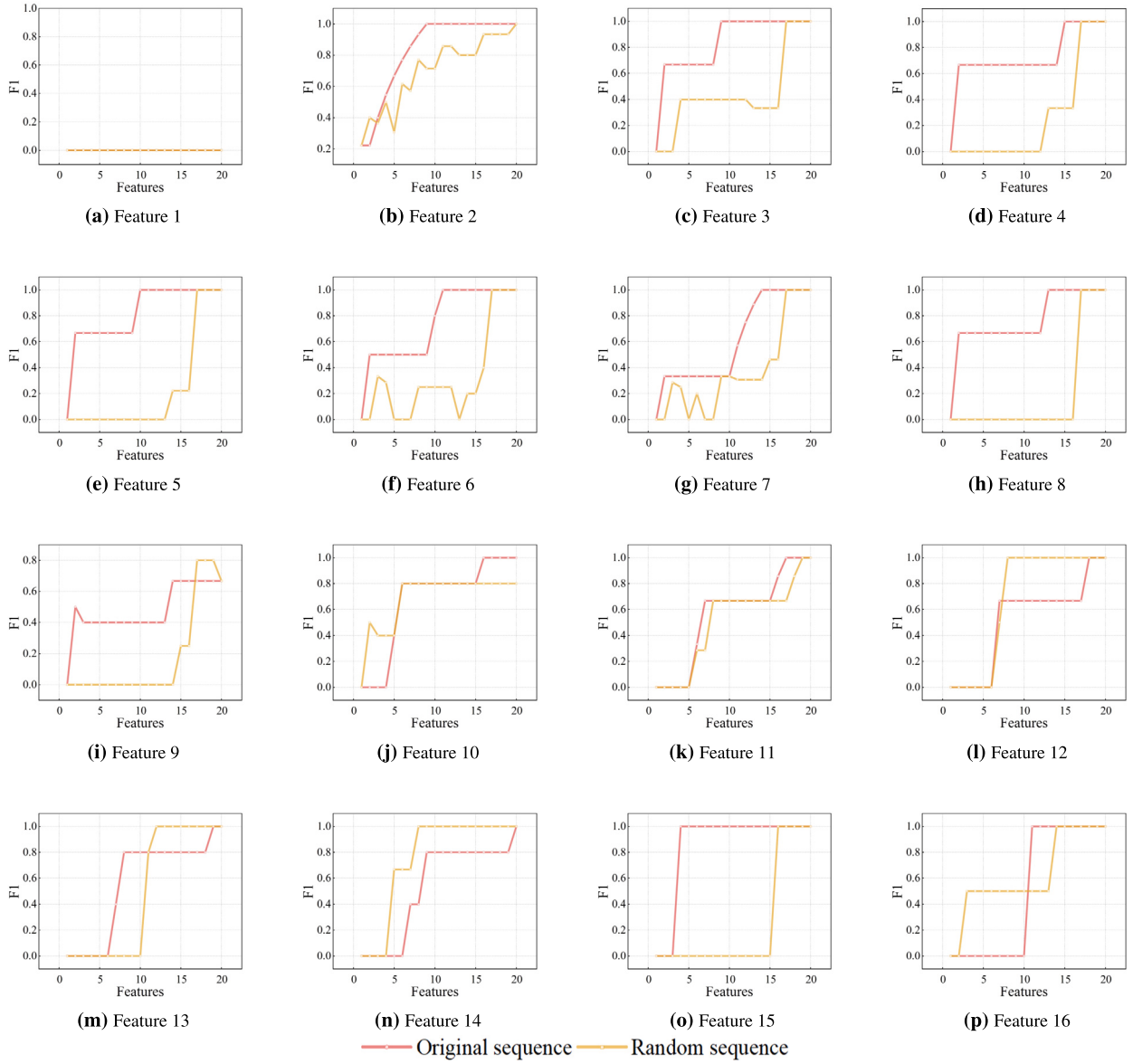


Fig. 11. Performance analysis with features arriving and sequence changes on Child dataset, the first 16 features as target variables, respectively. All 20 features can be seen in the supplementary file at the link https://github.com/youdianlong/LCSLSF.

5.7. A case study of real scenario

In this section, we perform a case study to verify the performance of $LCSL_{SF}$ and its rivals. The dataset in the case named the Lung Cancer Simple Set (LUCAS),[4] comes from a real scenario of a medical diagnosis problem, where the task is to identify patients with lung cancer given a set of socioeconomic and clinical factors of putative causal relevance. The LUCAS dataset consists of 2000 observations. The ground truth consists of 12 binary variables that include anxiety, peer pressure, day of birth, smoking, genetics, yellow fingers, lung cancer, attention disorder, cough, fatigue, allergy, car accidents, and their causal relations. There are no missing values in the dataset. As the LUCAS is generated artificially by causal BN with variables, the true nature of the underlying causal relationship is known. Hence we use a benchmark dataset to illustrate the effectiveness of the $LCSL_{SF}$.

Fig. 17 shows the local causal structure of the LUCAS dataset, where blue nodes represent features, the yellow node is the target variable, and the arcs represent causes and effects between nodes (e.g., the direct causes of Lung Cancer are Smoking and Genetics and the direct effects are Coughing and Fatigue). In the experiments, we obtained the local causal structure of the target variable

[4] The LUCAS dataset is available at http://www.causality.inf.ethz.ch/data/LUCAS.html.

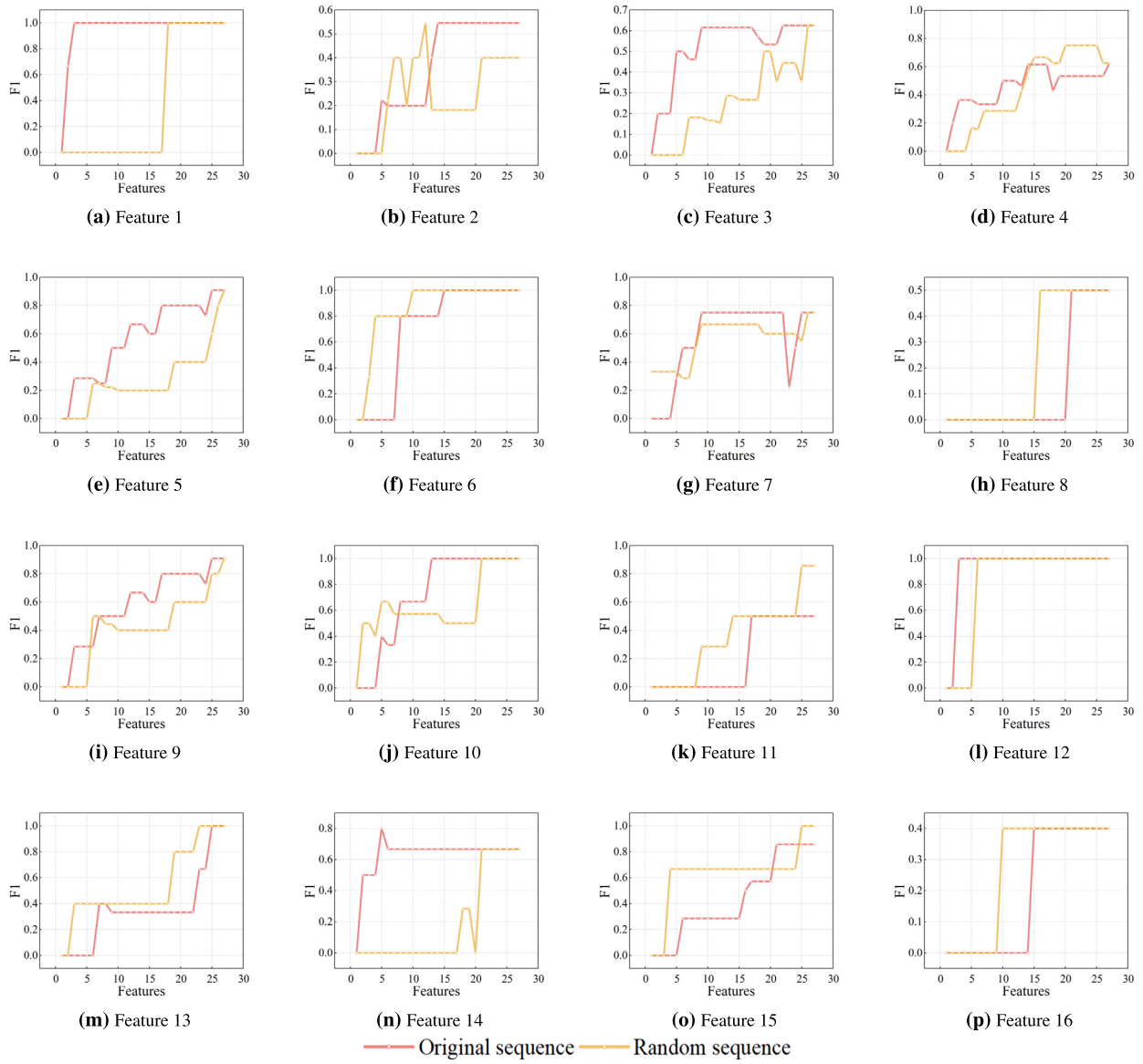


Fig. 12. Performance analysis with features arriving and sequence changes on Insurance dataset, the first 16 features as target variables, respectively. All 27 features can be seen in the supplementary file at the link https://github.com/youdianlong/LCSLSF.

"Lung Cancer" in the LUCAS dataset. Fig. 18 demonstrates the dynamic change of the local causal structure of lung cancer under the dynamic feature space. Table 16 documents the performance analysis of $LCSL_{SF}$ and other four LCSL algorithms. Due to the LUCAS dataset being continuous data, and as ELCS does not provide a method for processing continuous data, the experiment becomes unworkable. In terms of the causal structure, we find that $LCSL_{SF}$ obtains lower SHD, higher F1, Precision, and Recall on the LUCAS dataset. Therefore, compared to the real causal structure, $LCSL_{SF}$ achieves higher structural accuracy.

6. Conclusion

In this paper, we initially propose the $LCSL_{SF}$ algorithm to learn the local causal structure for streaming features, which is capable of more accurately mining an aMB including causal features by distinguishing the nodes of parents, children, and spouses from streaming features. Subsequently, we construct the local causal structure of the target variable with the direct cause/effects within aMB. Theoretical and empirical analyses demonstrate the performance of the proposed $LCSL_{SF}$ algorithms. In our experiments, we compare $LCSL_{SF}$ with five state-of-the-art LCSL methods and three variants. Extensive experimental results demonstrate that the proposed $LCSL_{SF}$ performs better than existing methods, especially for large-scale instances. To summarize for subsequent research, $LCSL_{SF}$ represents a typical LCSL method for static and dynamic feature space. From the experimental analyses of performance,

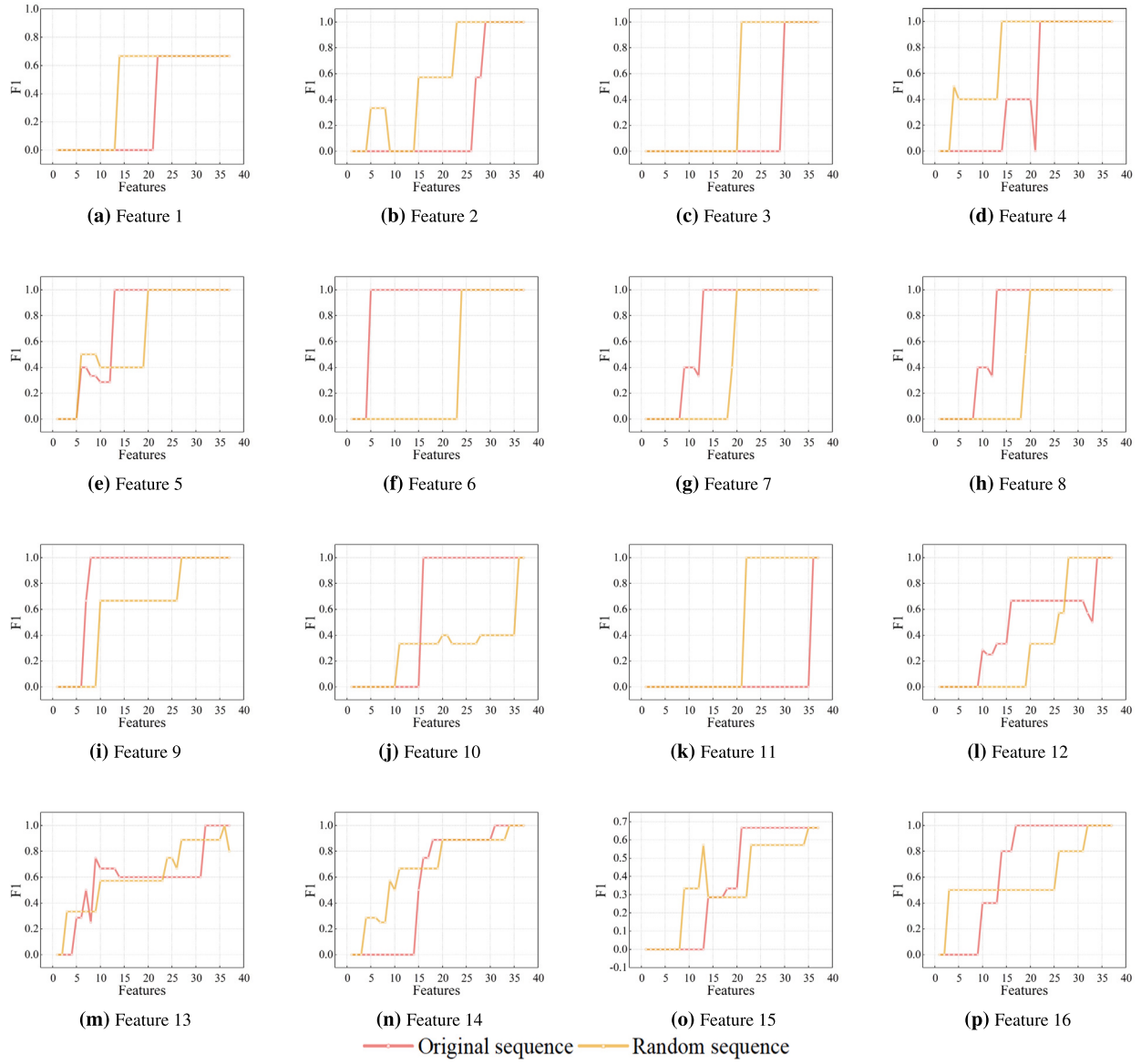


Fig. 13. Performance analysis with features arriving and sequence changes on Alarm dataset, the first 16 features as target variables, respectively. All 37 features can be seen in the supplementary file at the link https://github.com/youdianlong/LCSLSF.

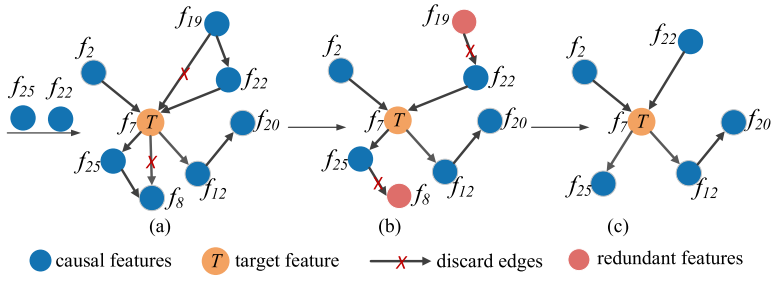


Fig. 14. Ascent performance when hidden causal features are received.

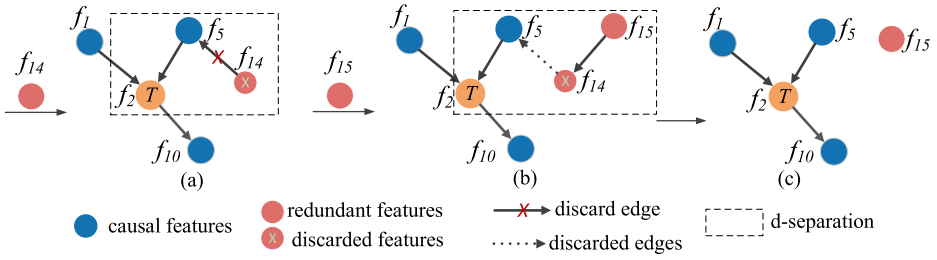


Fig. 15. Redundant features are unfiltered from the aMB due to the impact of hidden variables / not received features.

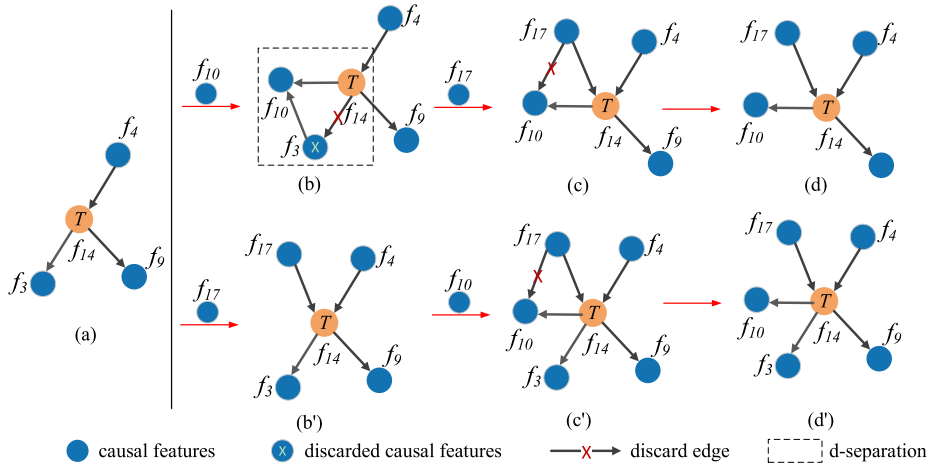


Fig. 16. Causal features are removed from the aMB due to the impact of hidden variables / not received features.

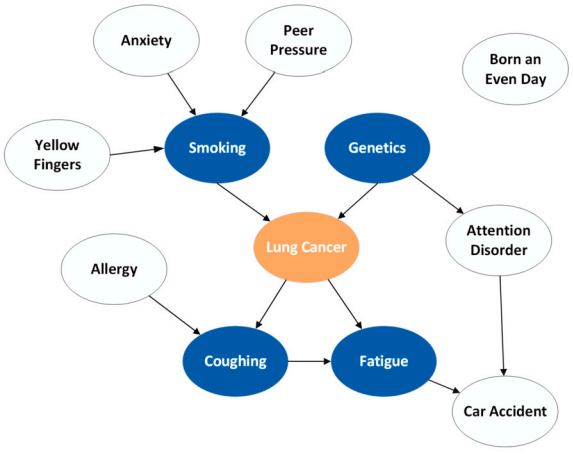


Fig. 17. The Bayesian Network of LUCAS.

statistics, and stability, $LCSL_{SF}$ has solved the traditional LCSL methods that are unsuitable for constructing the local causal structure from streaming features. Meanwhile, it provides methodological references for follow-up research.

Future research will include: 1) further improving the efficiency of $LCSL_{SF}$ to handle dynamic streaming features in real-time, and 2) mining local causal structures from arbitrary stream data with double-dimensional changes of streaming features/instances.

CRediT authorship contribution statement

Dianlong You: Conceptualization, Data curation, Formal analysis, Funding acquisition, Methodology, Validation, Writing – original draft, Writing – review & editing. **Siqi Dong:** Conceptualization, Data curation, Formal analysis, Methodology, Writing – original draft, Writing – review & editing. **Shina Niu:** Investigation, Resources. **Huigui Yan:** Investigation, Resources. **Zhen Chen:** Fund-

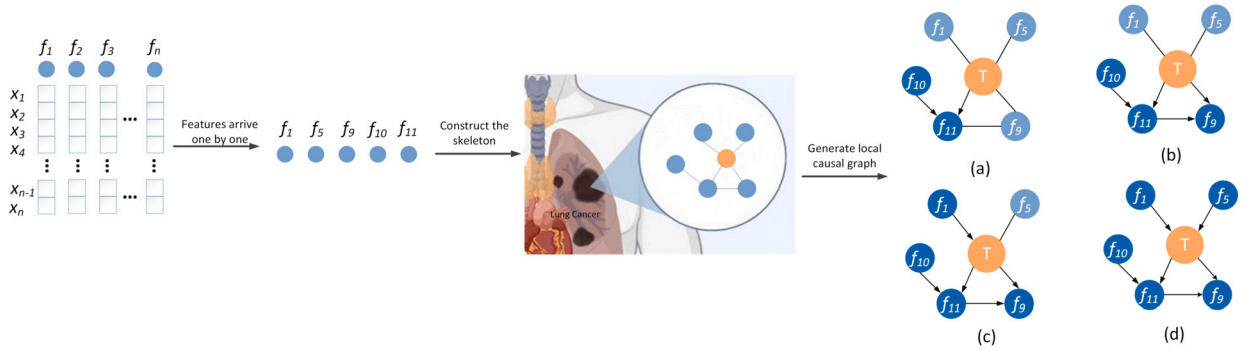


Fig. 18. A schematic diagram of mining the local causal structure from lung cancer data with dynamic feature space. f_1: Smoking; f_5: Genetics; f_9: Fatigue; f_{10}: Allergy; f_{11}: Coughing.

Table 16
Comparative evaluation among five algorithms.

Methond	SHD (↓)	F1(↑)	Precision (↑)	Recall (↑)
CMB	0.50	0.75	0.75	0.75
PCD-by-PCD	0.75	0.67	0.67	0.67
MB-by-MB	2.25	0.57	0.47	0.78
LCS-FS	0.83	0.58	0.61	0.65
$LCSL_{SF}$	**0.25**	**0.85**	**0.85**	**0.85**

ing acquisition, Resources. **Shunfu Jin:** Funding acquisition, Supervision. **Di Wu:** Funding acquisition, Resources. **Xindong Wu:** Methodology, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

This paper is partially supported by the grants from National Natural Science Foundation of China No.62276226, No.62176070, No.62102348, and No.62120106008; Natural Science Foundation of Hebei Province No.F2021203038, and No.F2022203012. Project of Hebei Key Laboratory of Software Engineering No.22567637H.

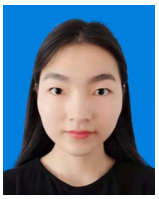
References

[1] K. Yu, X. Guo, L. Liu, J. Li, H. Wang, Z. Ling, X. Wu, Causality-based feature selection: methods and evaluations, ACM Comput. Surv. 53 (5) (2020) 1–36.
[2] M.J. Vowels, N.C. Camgoz, R. Bowden, D'ya like dags? A survey on structure learning and causal discovery, ACM Comput. Surv. 55 (4) (2021) 1–36.
[3] K. Yu, L. Liu, J. Li, A unified view of causal and non-causal feature selection, ACM Trans. Knowl. Discov. Data 15 (4) (2021) 1–46.
[4] R. Guo, L. Cheng, J. Li, P.R. Hahn, H. Liu, A survey of learning causality with data: problems and methods, ACM Comput. Surv. 53 (4) (2020) 1–37.
[5] B. Schölkopf, F. Locatello, S. Bauer, N.R. Ke, N. Kalchbrenner, A. Goyal, Y. Bengio, Towards causal representation learning, Proc. IEEE 109 (5) (2021) 612–634.
[6] X. Shen, S. Ma, P. Vemuri, G. Simon, Challenges and opportunities with causal discovery algorithms: application to Alzheimer's pathophysiology, Sci. Rep. 10 (1) (2020) 1–12.
[7] N.E. Fenton, S. McLachlan, P. Lucas, K. Dube, G.A. Hitman, M. Osman, E. Kyrimi, M. Neil, A Bayesian network model for personalised COVID19 risk assessment and contact tracing, MedRxiv (2021), 2020–2007.
[8] M. Badsha, A.Q. Fu, et al., Learning causal biological networks with the principle of Mendelian randomization, Front. Genet. 10 (2019) 460.
[9] Ö. Babur, A. Luna, A. Korkut, F. Durupinar, M.C. Siper, U. Dogrusoz, A.S.V. Jacome, R. Peckner, K.E. Christianson, J.D. Jaffe, et al., Causal interactions from proteomic profiles: molecular data meet pathway knowledge, Patterns 2 (6) (2021) 100–257.
[10] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT Press, 2009.
[11] N.K. Suchetha, A. Nikhil, P. Hrudya, Comparing the wrapper feature selection evaluators on Twitter sentiment classification, in: 2nd International Conference on Computational Intelligence in Data Science, ICCIDS 2019, February 21, 2019 - February 23, 2019, ICCIDS 2019 - 2nd International Conference on Computational Intelligence in Data Science, Proceedings, IEEE, Institute of Electrical and Electronics Engineers Inc., 2019, pp. 1–6.
[12] X. Wang, H. Ren, X. Guo, A novel discrete firefly algorithm for Bayesian network structure learning, Knowl.-Based Syst. 242 (2022) 108–426.
[13] I. Tsamardinos, L.E. Brown, C.F. Aliferis, The max-min hill-climbing Bayesian network structure learning algorithm, Mach. Learn. 65 (2006) 31–78.

[14] D. Margaritis, S. Thrun, Bayesian network induction via local neighborhoods, Adv. Neural Inf. Process. Syst. 12 (2000) 505–511.
[15] X. Zheng, B. Aragam, P. Ravikumar, E.P. Xing, Dags with no tears: continuous optimization for structure learning, Adv. Neural Inf. Process. Syst. 2018-December (2018) 9472–9483.
[16] Y. Wang, F. Cao, K. Yu, J. Liang, Local causal discovery in multiple manipulated datasets, IEEE Trans. Neural Netw. Learn. Syst. (2022) 1–13.
[17] X. Yu, W. Lam, Probabilistic joint models incorporating logic and learning via structured variational approximation for information extraction, Knowl. Inf. Syst. 32 (2) (2012) 415–444.
[18] J. Yin, Y. Zhou, C. Wang, P. He, C. Zheng, Z. Geng, Partial orientation and local structural learning of causal networks for prediction, in: Causation and Prediction Challenge, PMLR, 2008, pp. 93–105.
[19] C. Wang, Y. Zhou, Q. Zhao, Z. Geng, Discovering and orienting the edges connected to a target variable in a dag via a sequential local learning approach, Comput. Stat. Data Anal. 77 (2014) 252–266.
[20] T. Gao, Q. Ji, Local causal discovery of direct causes and effects, in: Proceedings of the 28th International Conference on Neural Information Processing Systems, vol. 2, NIPS'15, MIT Press, Cambridge, MA, USA, 2015, pp. 2512–2520.
[21] S. Yang, H. Wang, K. Yu, F. Cao, X. Wu, Towards efficient local causal structure learning, IEEE Trans. Big Data 8 (6) (2021) 1592–1609.
[22] Z. Ling, K. Yu, H. Wang, L. Li, X. Wu, Using feature selection for local causal structure learning, IEEE Trans. Emerg. Top. Comput. Intell. 5 (4) (2021) 530–540.
[23] Y. Lv, Y. Lin, X. Chen, D. Wang, C. Wang, Online streaming feature selection based on feature interaction, in: 2020 IEEE International Conference on Knowledge Graph (ICKG), IEEE, 2020, pp. 49–57.
[24] K. Yu, X. Wu, W. Hao, D. Wei, Causal discovery from streaming features, in: 2010 IEEE International Conference on Data Mining, 2010, pp. 1163–1168.
[25] X. Wu, K. Yu, W. Ding, H. Wang, X. Zhu, Online feature selection with streaming features, IEEE Trans. Pattern Anal. Mach. Intell. 35 (5) (2013) 1178–1192.
[26] D. You, R. Li, S. Liang, M. Sun, X. Ou, F. Yuan, L. Shen, X. Wu, Online causal feature selection for streaming features, IEEE Trans. Neural Netw. Learn. Syst. 34 (3) (2023) 1563–1577.
[27] K. Yu, X. Wu, W. Ding, J. Pei, Scalable and accurate online feature selection for big data, ACM Trans. Knowl. Discov. Data 11 (2) (2016) 1–39.
[28] J. Cai, J. Luo, S. Wang, S. Yang, Feature selection in machine learning: a new perspective, Neurocomputing 300 (5) (2018) 70–79.
[29] X. Wu, X. Xu, J. Liu, H. Wang, B. Hu, F. Nie, Supervised feature selection with orthogonal regression and feature weighting, IEEE Trans. Neural Netw. Learn. Syst. 32 (5) (2020) 1831–1838.
[30] J. Yang, A. Shen, K. Yu, Y. Chen, Predicting the semantic characteristics of pulmonary nodules using feature selection based on maximum-relevance minimum-redundancy, in: 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE, 2019, pp. 1318–1323.
[31] K. Yu, L. Liu, J. Li, W. Ding, T.D. Le, Multi-source causal feature selection, IEEE Trans. Pattern Anal. Mach. Intell. 42 (9) (2020) 2240–2256.
[32] I. Tsamardinos, C.F. Aliferis, Towards principled feature selection: relevancy, filters and wrappers, in: International Workshop on Artificial Intelligence and Statistics, PMLR, 2003, pp. 300–307.
[33] C.F. Aliferis, I. Tsamardinos, A. Statnikov, Hiton: a novel Markov blanket algorithm for optimal variable selection, AMIA Annual Symp. Proc. 2003 (2003) 21–25.
[34] H. Wang, Z. Ling, K. Yu, X. Wu, Towards efficient and effective discovery of Markov blankets for feature selection, Inf. Sci. 509 (2020) 227–242.
[35] Z. Ling, K. Yu, H. Wang, L. Liu, W. Ding, X. Wu, BAMB: a balanced Markov blanket discovery approach to feature selection, ACM Trans. Intell. Syst. Technol. 10 (5) (2019) 1–25.
[36] D. Wu, Y. He, X. Luo, M. Zhou, A latent factor analysis-based approach to online sparse streaming feature selection, IEEE Trans. Syst. Man Cybern. Syst. 52 (11) (2021) 6744–6758.
[37] P. Zhou, N. Wang, S. Zhao, Online group streaming feature selection considering feature interaction, Knowl.-Based Syst. 226 (2021) 107–157.
[38] M. Scanagatta, A. Salmerón, F. Stella, A survey on Bayesian network structure learning from data, Prog. Artif. Intell. 8 (4) (2019) 425–439.
[39] Y. Jiang, Z. Liang, H. Gao, Y. Guo, Z. Zhong, C. Yang, J. Liu, An improved constraint-based Bayesian network learning method using Gaussian kernel probability density estimator, Expert Syst. Appl. 113 (2018) 544–554.
[40] R. Bhattacharya, T. Nagarajan, D. Malinsky, I. Shpitser, Differentiable causal discovery under unmeasured confounding, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 2314–2322.
[41] T. Niinimaki, P. Parviainen, Local structure discovery in Bayesian networks, in: Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence, Catalina Island, CA, USA, 14–18 August, 2012, pp. 634–643.
[42] I. Tsamardinos, C.C.F. Aliferis, A.A. Statnikov, Time and sample efficient discovery of Markov blankets and direct causal relations, in: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD-2003, 2003, pp. 673–678.
[43] T. Gao, K.P. Fadnis, M. Campbell, Local-to-global Bayesian network structure learning, in: Proceedings of the 34th International Conference on Machine Learning, ICML, Sydney, NSW, Australia, 6–11 August, 2017, pp. 1193–1202.
[44] X. Zheng, B. Aragam, P. Ravikumar, E.P. Xing, Dags with no tears: continuous optimization for structure learning, in: 32nd Conference on Neural Information Processing Systems, NeurIPS 2018, December 2, 2018 - December 8, 2018, in: Advances in Neural Information Processing Systems, Neural Information Processing Systems Foundation, vol. 2018-December, 2018, pp. 9472–9483.
[45] M. Zhang, S. Jiang, Z. Cui, R. Garnett, Y. Chen D-vae, A variational autoencoder for directed acyclic graphs, in: 33rd Annual Conference on Neural Information Processing Systems, NeurIPS 2019, December 8, 2019 - December 14, 2019, in: Advances in Neural Information Processing Systems, Neural Information Processing Systems Foundation, vol. 32, 2019, pp. 1586–1598.
[46] Silverstein Craig, Brin Sergey, Motwani Rajeev, Jeff Ullman, Scalable techniques for mining causal structures, Data Min. Knowl. Discov. 4 (2) (2000) 163–192.
[47] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Series in Representation and Reasoning, 1988.
[48] P. Spirtes, C. Glymour, R. Scheines, Causation, Prediction, and Search, vol. 83, MIT Press, 2000.
[49] I. Guyon, A. Satnikov, C. Aliferis, Time series analysis with the causality workbench, in: NIPS Mini-Symposium on Causality in Time Series, PMLR, 2011, pp. 115–139.
[50] I. Guyon, S. Gunn, A.B. Hur, G. Dror, Design and Analysis of the Nips2003 Challenge, vol. 207, Springer Berlin Heidelberg, 2006, pp. 237–263.

Associate professor **Dianlong You** is currently an Associate Professor, received the Ph.D. degree in computer application technology from Yanshan University, Qinhuangdao, HeBei, China, in 2014. From 2017-8 to 2018-8, he was a visiting scholar with the School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA, USA. From 2022-8 to 2023-4, he was a visiting scholar with the Data Science Institute, University of Technology Sydney, Sydney, NSW, AUS. His current research interests include machine learning, streaming feature selection and causal discovery. He has over 20 publications including journals of IEEE TNNLS, IEEE TKDE, IEEE TKDD, INS, and KBS, etc. Dr. You is a member of IEEE.

Siqi Dong is currently a Master Student in the School of Information Science and Engineering, Yanshan University, Qinhuangdao, HeBei. Her current research interests include streaming feature selection, and causal discovery.



Shina Niu is currently a Master Student in School of Information Science and Engineering, Yanshan University, Qinhuangdao, HeBei. Her current research interests are focused on causal representation learning, counterfactual inference and causal discovery.

Huigui Yan is currently a Ph.D. student in the School of Information Science and Engineering, Yanshan University, Qinhuangdao, HeBei. His current research interests include streaming feature selection and causal discovery.

Associate professor **Zhen Chen** received his Ph.D. and B.S. in computer science and technology from Yanshan University in China, in 2017 and 2010, respectively. He is currently working on service computing and data mining.

Professor **Shunfu Jin** received the B.Eng. degree in computer and application from the North East Heavy Machinery College, Qiqihaer, China, the M.Eng. degree in computer science and the Dr.Eng. degree in circuit and system from Yanshan University, Qinhuangdao, China. She is currently a Professor at the School of Information Science and Engineering, Yanshan University. Her research interests include artificial intelligence, performance evaluation of communication networks and queueing systems. She has over 20 publications including journals of Performance Evaluation and Future Generation Computer Systems, etc.

Professor **Di Wu** (Member, IEEE) received his Ph.D. degree from the Chongqing Institute of Green and Intelligent Technology (CIGIT), Chinese Academy of Sciences (CAS), China in 2019 and then joined CIGIT, CAS, China. He is currently a Professor of the College of Computer and Information Science, Southwest University, Chongqing, China. He has over 70 publications, including 17 IEEE TRANSACTIONS papers and several conference papers on AAAI, ICDM, WWW, IJCAI, etc. He is serving as an Associate Editor for the Neurocomputing and Frontiers in Neurorobotics. His research interests include machine learning and data mining. His homepage: https://wudi1986.github.io/Homepage/.

Professor **Xindong Wu** (F'11) received his Bachelor's and Master's degrees in Computer Science from the Hefei University of Technology, China, and his Ph.D. degree in Artificial Intelligence from the University of Edinburgh, Britain, in 1993. He is currently the Director and Professor of the Key Laboratory of Knowledge Engineering with Big Data (the Ministry of Education of China), Hefei University of Technology, China, and also a Senior Research Scientist in the Research Center for Knowledge Engineering at Zhejiang Lab, China. He is a Foreign Member of the Russian Academy of Engineering, and a Fellow of IEEE and the AAAS. He is the Steering Committee Chair of ICDM and the Editor in-Chief of KAIS, TKDE between 2005 and 2008 and Co-Editor-in-Chief of TKDD between 2017 and 2020. His research interests include big data analytics, data mining and knowledge engineering.